Fachhochschule Köln
Cologne University of Applied Sciences

Gestenerkennung mit der SFA

Klassifizierung von beschleunigungsbasierten 3D-Gesten des Wii-Controllers

Masterthesis

ausgearbeitet von

Kristine Hein
kristine.hein@gm.fh-koeln.de

zur Erlangung des akademischen Grades
Master of Science (M.Sc.)

vorgelegt an der
Fachhochschule Köln
Campus Gummersbach
Fakultät für Informatik und
Ingenieurwissenschaften

im Studiengang
Medieninformatik

Erster Prüfer: Prof. Dr. Wolfgang Konen
Fachhochschule Köln

Zweiter Prüfer: Prof. Dr. Horst Stenzel
Fachhochschule Köln

Gummersbach, im Juli 2010

Kurzfassung: Diese Arbeit untersucht die Slow Feature Analysis (SFA) auf ihre Möglichkeiten zur Gestenerkennung auf Basis von gerätebasierten 3D-Beschleuningungsdaten des Wii-Controllers. Bisherige Ansätze zur Klassifizierung von beschleunigungsbasierten 3D-Gesten verwenden häufig ein HMM-basiertes Verfahren. Die Erkennung von Anfang und Ende einer Geste erfolgt in der Regel entweder anhand eines Ruhelage-Filters oder mit Hilfe von externen Markierungen der Gesten bei ihrer Erhebung. Hier soll nun ein anderer Ansatz, nämlich die Gestenerkennung mit der Slow Feature Analysis (SFA) vorgestellt, näher untersucht und mit anderen gängigen Verfahren verglichen werden. Die SFA ist ein Lernalgorithmus, der die am langsamsten variierenden Signale aus einem sich schnell ändernden Eingangssignal findet, und anhand dessen lernt auf die Gestenklassen zu schließen. Die SFA wurde bereits erfolgreich zur Mustererkennung von handgeschriebene Ziffern eingesetzt und zeigt auch in diesem Ansatz für die Gestenerkennung vergleichbare Ergebnisse mit anderen gängigen Klassifizierungsverfahren. Zur Segmentierung eines Gestensignal um Anfangs- und Endpunkt einer Geste zu ermitteln, wurden unterschiedliche Varianten mit der SFA und ein anderer alternativer regelbasierter Ansatz untersucht. Diese lieferten vergleichbare Ergebnisse mit einem gängigen dynamischen Segmentierungsverfahren.

Inhaltsverzeichnis

1. Einleitung **4**

2. Klassifizierung von Gesten **6**
2.1. State of the art . . . 6
2.2. Slow Feature Analysis (SFA) . . . 8
2.3. Signalerfassung und Datenaufbereitung . . . 11
2.3.1. Auslesen und Aufzeichnen der Wiimote Gestendaten . . . 11
2.3.2. Verwendete Daten . . . 15
2.3.3. Vorverarbeitung . . . 17
2.4. Klassifizierungstests . . . 19
2.4.1. Durchführung der Untersuchungen . . . 19
2.4.2. Test I - Auswirkung der Reduktion durch die PCA auf das Ergebnis . . . 22
2.4.3. Test II - Parametric Bootstrap . . . 24
2.4.4. Test III - Erkennung der Gesten personenabhängig . . . 29
2.4.5. Test IV - Erkennung der Gesten personenübergreifend . . . 32
2.4.6. Test V - Erkennung der Gesten einer neuen Person . . . 34
2.4.7. Test VI - Minimales Trainingsset . . . 38
2.5. Diskussion . . . 40

3. Segmentierung von Gestensignalen **42**
3.1. State of the art . . . 42
3.2. Signalerfassung und Datenaufbereitung . . . 44
3.2.1. Verwendete Daten . . . 44
3.2.2. Kalibrierung . . . 47
3.3. Segmentierungstests . . . 49
3.3.1. Regelbasierte Segmentierung . . . 49
3.3.2. Segmentierung mit SFA . . . 50
3.3.3. Segmentierung über durchschnittliche Bewegungsänderung . . . 61
3.4. Vergleich und Bewertung . . . 64

4. Fazit **68**

A. Anhang **69**
A.1. Entstandene Artefakte . . . 69
A.2. Ergebnisse des Klassifikationstest III mit HMM . . . 70
A.3. Regelbasierte Segmentierung (Hysterese) . . . 71
A.4. Ergebnisse zur Voruntersuchung: Segmentierung einfache SFA mit Beschleunigungsbetrag . . . 73
A.5. Ergebnisse zur Voruntersuchung: Segmentierung einfache SFA mit Beschleunigungsrohdaten . . . 75
Darstellung des SFA-Segmentierung mit Inputs und Outputs . . . 75
Darstellung des SFA Outputs mit Median . . . 77

1. Einleitung

Moderne Techniken eröffnen neue Möglichkeiten der Mensch-Computer-Interaktion. Durch Interaktionsgeräte, wie die Wiimote[1], die mit Beschleuningungssensoren ausgestattet sind, lassen sich andere Kommunikationskanäle nutzen, als dies bisher mit herkömmlichen Ein- und Ausgabegeräten möglich war. Diese Kommunikationskanäle bieten auch die Chance, neue Kommunikationsmodalitäten zu verwenden. Beispielsweise lassen sich Gesten als zusätzliche Eingabemodalität nutzen, die zur Steuerung von Maschinen eingesetzt werden kann.

Gesten sind Bewegungen, die eine Kommunikation in vielerlei Hinsicht unterstützen können. Geräte, die diese Art der Kommunikation bieten, können eine Interaktion mit Maschinen auf natürlichere Weise ermöglichen, als ohne Gestenunterstützung. Da aber keine Geste der anderen gleicht, muss eine Möglichkeit gefunden werden, solche nicht eindeutigen Muster wiederzuerkennen. In Disziplinen wie Computer Vision und maschinellem Lernen werden seit einigen Jahren bereits unterschiedliche Techniken und Verfahren zur Gestenerkennung erforscht. Doch gerade die aktuelle Verbreitung von Beschleunigungssensoren in handelüblichen Geräten und die Entwicklung neuer Lern-Verfahren geben den Anstoß, diese beiden Aspekte ebenfalls auf ihre Möglichkeiten zu Gestenerkennung zu untersuchen. In dieser Arbeit soll deswegen die Erkennung von beschleunigungsbasierten 3D-Gesten mit einer für die Gestenerkennung bisher nicht angewandten Methode, der Slow Feature Analysis (SFA), untersucht werden.

Die SFA ist ein Lern-Verfahren, das aus den Neurowissenschaften stammt und 1998 von Laurenz Wiskott [Wis98] erstmalig vorgestellt wurde. Dieses ermöglicht es, aus sich schnell verändernden Signalen die langsamsten Merkmale herauszufiltert.
Im Bereich der Mustererkennung wurde die SFA bereits erfolgreich zur Klassifizierung eingesetzt und zur Wiedererkennung von handgeschriebenen Ziffern verwendet. Im Fall der Gestenerkennung soll die SFA in dieser Arbeit daraufhin untersucht werden, inwieweit sie sowohl zum Klassifizieren einzelner Gestenklassen eingesetzt werden kann, als auch zum Auffinden von Anfang und Ende der Gestenmuster in einem kontinuierlichen Datenstrom.

Die für diese Arbeit erhobenen Gestendaten sind mehrdimensionale Zeitreihen, die aus den Beschleunigungsdaten der Wiimote bestehen. Diese Beschleunigungsdaten werden kontinuierlich von einem Rechner, mit dem die Wiimote verbunden ist, ausgelesen, während mit der Wiimote in der Hand die Gesten ausgeführt werden. Aus diesen Zeitreihen soll zunächst mittels Segmentierung der Anfang und das Ende einer Geste gefunden werden. Hierfür ist es notwendig, die Eigenschaften einer Geste zu betrachten, um herauszufinden, wie sich diese Geste zu einer relativen Ruhelag vor und nach der Ausführung abgrenzt. Eine Geste besteht nach Kendon [Ken72] aus drei Phasen: der Vorbereitung, der eigentlichen Durchführung und der Rückführung in die Ruheposition. Um Gesten in Form von Bewegung messbar und klassifizierbar zu machen, werden sie in dieser Arbeit über ihre räumlichen und zeitlichen Merkmale gemessen. Anhand dieser Merkmale läßt sich angelehnt an [Pre08] eine Geste wie folgt definieren: “Eine Geste

- beginnt mit einer schnellen (oder kurzfristigen) Beschleunigung,
- ändert während des Verlauf kontinuierlich die Richtung,
- endet in einer nahezu ruhigen Position und
- dauert mehr als 0.4 Sekunden.“

[1] Wii-Controller der Nintendo-Spielkonsole

Durch eine Abgrenzung von Gesten-Segmenten zu Nicht-Gesten-Segmenten, also die Abschnitte in denen sich die Wiimote in relativer Ruhelage befindet, lassen sich die eigentlichen Gestenbereiche rausfiltern. Diese Gestenmuster sollen anschließend der richtigen Klasse eines definierten Gestensets zugeordnet werden.

Die SFA wird in dieser Arbeit sowohl im Bereich der Klassifizierung als auch im Bereich der Segmentierung eingesetzt, allerdings auf unterschiedliche Art und Weise. Diese Arbeit gliedert sich deswegen in zwei Hauptkapitel: den ersten Teil, der die Klassifizierung mit der SFA näher untersucht und den zweiten Teil, der die Segmentierung betrachtet.

Im Kapitel "Klassifizierung" wird die Frage beantwortet, ob und wie gut die SFA unter unterschiedlichen Voraussetzungen zur Klassifizierung von Gesten geeignet ist und ob sie sich mit anderen gängigen Verfahren im Bereich der Gestenerkennung vergleichen läßt.

Im Kapitel "Segmentierung" werden mehrere Möglichkeiten zur Segmentierung mit der SFA untersucht, aber auch zwei andere Ansätze. Diese Ansätze werden abschließend einander gegenübergestellt.

2. Klassifizierung von Gesten

In diesem Kapitel werden zunächst gängige Ansätze zur Gestenerkennung von beschleunigungsbasierten 3D-Gesten vorgestellt. Anschließend wird das grundlegende Verfahren der SFA beschrieben und wie sich diese zur Mustererkennung einsetzen lässt.
Ausgehend von der Datenerhebung bis zu den eigentlichen Klassifikationstests waren einige Schritte notwendig, um die Gestenerkennung mit der SFA testen zu können:

- Erhebung und Auswahl von Daten (inklusive der Implementierung einer Java-Anwendung zur Erhebung der Daten)
- Vorverarbeitung der Daten
- Untersuchungen zur Erzeugung von künstlichen Trainingsdaten aus den Originaldaten
- Implementierung der Testverfahren in der jeweiligen Testumgebung (SFA in Matlab, HMM in der Java-Anwendung)

Nach der Beschreibung der vorbereitenden Schritte werden die einzelnen Untersuchungen zur Klassifizierung mit der SFA und ihre Ergebnisse dargestellt. In Fällen, in denen ein direkter Vergleich mit anderen Klassifizierungsverfahren möglich war, werden deren Ergebnisse denen der SFA direkt gegenübergestellt. Abschließend wird das Gesamtergebnis der Gestenklassifizierung mit der SFA diskutiert.

2.1. State of the art

Ein häufig verwendetes Verfahren zur Klassifizierung beschleunigungsbasierter 3D-Gesten ist das Hidden Markov Modell (HMM). Dieses Modell verwendet eine stochastische Signalmodellierungsmethode, die den Markovprozess erweitert [MKKK04]. Es generiert dabei für alle Zeitpunkte eine Zustandsmenge, in der jeder Zustand zu einem physikalisch beobachtbaren Ereignis gehört. Es beinhaltet dabei den Fall, dass das Ergebnis eines Ereignisses mit einer gewissen Wahrscheinlichkeit zu einer definierten Zustandsmenge gehört. Dieses Modell wird häufig sowohl zur Spracherkennung als auch zur Gestenerkennung in unterschiedlichsten Bereichen eingesetzt. Zur reinen Klassifizierung beschleunigungsbasierter 3D-Gesten wird das HMM mit Eingabegeräten wie Mobiltelefonen bei Prekopcsak[Pre08] oder einem Datenhandschuh bei Hofmann[HHH98] eingesetzt.
Von Mäntyjärvi et al.[MKKK04] wird das HMM ebenfalls eingesetzt. Die dort verwendeten Gesten werden zur Steuerung von DVD-Playern verwendet. Erkannt und gesteuert werden die Gesten dabei über ein selbst konstruiertes Gerät, das mit entsprechenden Beschleunigungssensoren ausgestattet ist. Zum Training des Modells werden nur sehr wenige Benutzereingaben benötigt, da aus vorhandenen Gesten durch Verrauschen[1] zusätzliche Trainingsdaten erzeugt werden.
Ein Ansatz, die Wiimote zur Gestenerkennung einzusetzen, wird von Malmestig und Sundberg in [MS08] beschrieben, die ebenfalls ein HMM nutzen, um Gebärden wieder zu erkennen.
Schlömer et al. [SPHB08] haben ein javabasiertes Framework namens WiiGee [PS07] entwickelt, das ebenfalls auf Grundlage des HMM-Verfahrens Gesten der Wiimote klassifizieren kann. Dieses Framework wird in dieser Arbeit näher betrachtet. Es dient als Basis zur Implementierung von HMM-Vergleichstests.
Allen genannten Ansätzen mit HMM ist gemein, dass sie keine Aussage darüber treffen, ob und wie gut eine personübergreifende Klassifizierung von Gesten oder die Erkennung von Gesten einer neuen Person mit ihren Ansätzen funktioniert.

[1] Veränderung des Signals durch einen Störfaktor.

Liu et al. [LZWV09] präsentieren mit uWave einen anderen interessanten Ansatz zur Gestenerkennung mit der Wiimote, in dem sie zwar ebenfalls vorwiegend personenabhängige Tests durchführen, allerdings zeigen sie auch, dass selbst Gesten von ein und der selben Person an verschiedenen Tagen unterschiedlich ausgeführt werden. Dies verschlechtert die Ergebnisse für eine Erkennung mit dem von ihnen gewählten Ansatz. Sie zeigen aber auch, dass für die Erkennung von Gesten einer neuen Person große Trainingssets und statistische Methoden notwendig sind, um die personenübergreifenden Ähnlichkeitsmerkmale in den Gesten zu finden. Für die Erkennung werden die Daten ihres Ansatzes über nicht lineare Quantisierungsstufen quantisiert und anschließend die Ähnlichkeit der resultierenden Zeitreihen mit DTW (dynamic time warping) verglichen.
Ansätze wie WiiGLE von Rehm et al.[RBA08], die ebenfalls die Wiimote als Eingabegeräte für die Gestenerkennung verwenden, setzen auf "simple" Klassifizierer wie Nearest Neighbor (NN), Naive Bayes (NB) oder Multilayer Perceptron (MLP) und vergleichen diese auch mit HMM. Für ihr gewähltes Verfahren zur Erkennung von kulturabhängigen Gesten sind die simplen Klassifizierer ausreichend (und HMM ist nicht nötig). Die Gesten werden dabei durch die Energie des Signals, die Ausführungsgeschwindigkeit und die räumliche Ausdehnung einer Geste charakterisiert.
Der Ansatz, der von Nintendo selbst verfolgt wird, ist die Verwendung künstlicher Neuronaler Netze (ANNs) zur Klassifizierung der Gesten [MS08].

2.2. Slow Feature Analysis (SFA)

Die Slow Feature Analysis (SFA) stammt ursprünglich als Lernalgorithmus aus den Neurowissenschaften und wurde entwickelt, um unüberwacht Invarianzen in visuellen Systemen zu finden [Wis98]. Eine Hauptaufgabe von Lernalgorithmen ist die Klassifizierung.
Dabei lernt die SFA in einem One-Shot-Verfahren, das heißt, sie wird einmal angelernt und wird dann in dieser Art für weitere Tests verwendet.
Die SFA ist ein unüberwachter Lernalgorithmus, der aus einem gegebenen Set von Eingangsfunktionen oder Eingangsvektoren die am langsamsten variierenden, aber dennoch relevanten Merkmale herausfiltert. Die SFA basiert dabei auf einer nicht-linearen Expansion des Eingangssignals und der Anwendung der Hauptkomponentenanalysis (PCA) auf dieses expandierte Signal sowie auf ihre zeitlichen Ableitungen. Sie kann dabei ein recht großes Set von dekorrelierten Merkmalen, die über ihren Index geordnet sind, lernen (Index 1 = langsamstes Signal bzw. Merkmal).

Die SFA nach [WS02], [Wis03] ist allgemein wie folgt definiert: Für eine N-dimensionale Zeitreihe als Eingangssignal, $\vec{x}(t) = (x_1(t), ..., x_N(t))$, soll ein Set von reelwertigen Ausgangsfunktionen $g_1(\vec{x}), g_2(\vec{x}), ..., g_M(\vec{x})$ gefunden werden, so dass jede Ausgangsfunktion $y_j(t) = g_j(\vec{x}(t))$ sich minimal über die Zeit t verändert:

$$\text{Zielfunktion: } \langle \dot{y_j}^2 \rangle_t \text{ sei minimal} \tag{2.1}$$

unter den Bedingungen:

$$\langle y_j \rangle_t = 0 \text{ (mittelwertfrei)} \tag{2.2}$$

$$\langle y_j^2 \rangle_t = 1 \text{ (Varianz von eins)} \tag{2.3}$$

$$\langle y_k y_j \rangle_t = 0 \text{ (Dekorrelation für } k > j) \tag{2.4}$$

wobei die spitzen Klammern $\langle \cdot \rangle_t$ den zeitlichen Mittelwert bezeichnen und $\dot{y}$ die zeitliche Ableitung.
Die Bedingungen (2.2) und (2.3) verhindern eine konstante Lösung. Bedingung (2.4) verhindert, dass verschiedene Dimensionen des Ausgangssignals dieselbe Information repräsentieren und ordnet dabei die Dimensionen, so dass die Komponenten mit der geringsten zeitlichen Veränderung am Anfang des Vektors stehen.

Schritte der SFA:

1. Normalisierung der Eingangsdaten, so dass sie mittelwertfrei sind und die Varianz 1 beträgt.

2. Nichtlineare Expansion des Eingangssignals (der expandierte Vektor $\vec{h}$ besteht dann aus Monomen [2] bis zum Grad d von $\vec{x}$). Bei quadratischer SFA (auch SFA^2, SFA2), wie sie in dieser Arbeit verwendet wird, sind das alle Monome ersten und zweiten Grades der Eingangsdimensionen[3]. Dadurch steigt die Dimensionalität der Daten hier von N auf $N + \frac{N(N+1)}{2}$. Mit weiter steigendem Grad steigt auch die Dimensionalität exponentiell.

3. Erneute Normalisierung des expandierten Signals, so dass der zeitliche Mittelwert 0 ist und die Kovarianzmatrix der Einheitsmatrix entspricht. Dieser Vorgang wird als Sphering oder Whitening bezeichnet.

[2] Monome sind Produkte von Potenzen der Variablen.
[3] Beispiel: expandierter Vektor h mit Monomen 2.Grades bei einem dreidimensionalen Eingangsvektor, $h = \{x_1^1, x_1^2, x_1x_2, x_1x_3, x_2^1, x_2^2, x_2x_3, x_3^1, x_3^2\}$

4. Auf die Kovarianzmatrix der zeitlichen Ableitung dieses Signals wird die Hauptkomponentenanalyse (PCA) [Pre07] angewandt, wobei die Eigenvektoren mit den niedrigsten Eigenwerten die Gewichtungsvektoren für die Berechnung der am langsamsten variierenden Funktionen darstellen.

Um diese Funktionen auf Testdaten anzuwenden, werden die Schritte 1 bis 3 durchlaufen, wobei die Normalisierung stets mit den aus den Trainingsdaten ermittelten Werten stattfindet. Danach werden die so expandierten und normalisierten Daten mit der Matrix W multipliziert, die aus den Eigenvektoren besteht, die zuvor mit der SFA ermittelt wurden.

$$y(t) = W * x'(t) \tag{2.5}$$

Dabei ist $x'(t)$ das expandierte und normalisierte Eingangssignal. W ist die Matrix der J Eigenvektoren mit den niedrigsten Eigenwerten, wobei der Eigenvektor mit dem niedrigsten Eigenwert an erster Stelle steht. $y(t)$ ist das resultierende, J-dimensionale Signal [Neu09].

Für Eingangsfunktionen, die über die Zeit betrachtet werden, wird die SFA wie zuvor beschrieben eingesetzt. Sie wird in dieser Arbeit für den Teil der Segmentierung von Gesten so verwendet. Für den Teil der Gestenerkennung, wie sie auch Berkes [Ber05] zur Klassifizierung von Ziffern verwendet hat, wird die Vorgehensweise zur Nutzung der SFA ein wenig verändert. Die Eingangsdaten für die SFA bestehen nicht wie zuvor aus den Zeitsignalen, sondern repräsentieren die einzelnen Pattern in Form von Merkmalsvektoren. Berkes formuliert die Zielfunktion für K-Klassen dahingehend, dass die Differenz zwischen zwei Merkmalsvektoren der selben Klasse minimal werden soll:

$$\text{Zielfunktion: } \frac{1}{n_{pair}} \cdot \sum_{m=1}^{K} \sum_{\substack{k,l=1,\\ k<l}}^{N_m} \left(g_j(p_k^{(m)}) - g_j(p_l^{(m)}) \right)^2 \text{ sei minimal} \tag{2.6}$$

p_k und p_l stellen dabei unterschiedliche Pattern einer Klassse dar. Durch diese Zielfunktion werden quasi alle möglichen "Mini-Zeitreihen" erstellt, die sich aus den Pattern derselben Klasse bilden lassen. N_m entspricht dabei der Anzahl der Pattern einer Klasse und n_{pair} ist die Anzahl aller möglichen Paarkombinationen derselben Klasse summiert über alle Klassen. Die Merkmalsvektoren werden zu Beginn durch eine PCA in ihrer Dimension reduziert.

Wie Berkes [Ber05] gezeigt hat, haben die K-1 langsamsten SFA Ausgangssignale erwartungsgemäß eine niedrige Varianz innerhalb einer Klasse, aber eine hohe Varianz zwischen unterschiedlichen Klassen. Die Ausgangswerte einer Klasse gruppieren sich um einen Zentralwert, wie in Abb. 2.1 zu sehen ist.

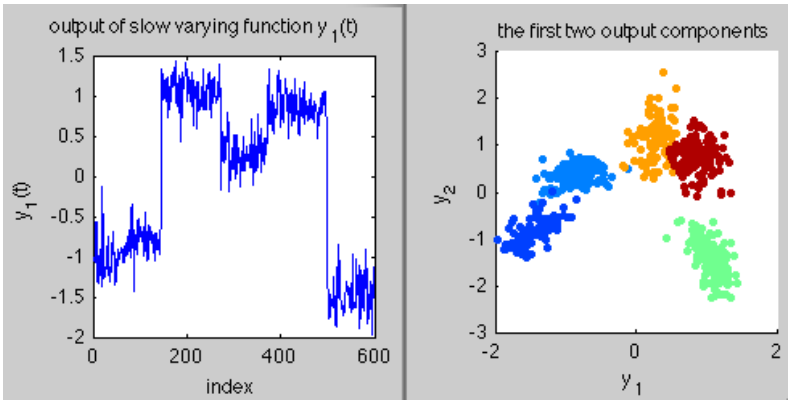

Abb. 2.1.: *Inter- und Intraclass Varianz für erstes SFA Ausgangssignal (links) und für erste/zweites SFA Ausgangssignal (rechts)*

Aus diesem Grund wird im Trainingsprozess ein Standard-Gaussklassifizierer mit den K-1 langsamsten SFA Ausgangssignalen trainiert. Der Gaussklassifizierer ist ein Ähnlichkeitsschätzer, der nach einer optimalen Position (also die minimale Distanz zwischen den Ausgangssignalen einer Klasse) sucht und daraufhin für jede Klasse in diesem (K-1)-dimensionalen Raum eine Gaussfunktion erstellt.

Dieser Ansatz lässt sich auch für die Gestenerkennung anwenden. Dazu werden als Trainingspattern die vorverarbeiteten und konkatenierten Werte der Beschleunigungssensoren der Wiimote verwendet. Diese Vorverarbeitung ist in 2.3.3 näher beschrieben. Die Bedingungen (2.2-2.4) können dadurch erfüllt werden, dass hier nicht der Durchschnitt über die Zeit genommen wird, sondern der Durchschnitt über alle Pattern.

In einem Matlab-Toolkit stellt Berkes eine SFA-Implementierung zur Verfügung [Ber03], deren Erweiterung, nämlich dem SFA-TK 2.6 von W.Konen [Kon09], in dieser Arbeit verwendet wird.

2.3. Signalerfassung und Datenaufbereitung

In diesem Abschnitt werden die Vorbereitungsschritte beschrieben, die notwendig waren, um Gestendaten zur Klassifizierung verwenden zu können.

2.3.1. Auslesen und Aufzeichnen der Wiimote Gestendaten

Zunächst werden im Folgenden die Technik und notwendige Softwaretools beschrieben, mit denen die Gesten aufgezeichnet wurden. Anschließend wird auf die verwendeten Gesten, die eigentliche Gestenaufzeichnung mit unterschiedlichen Probanden und die Speicherung der Daten näher eingegangen.

Technik und Tools

Technik Als Eingabegerät wurde die Wiimote verwendet, die drei Beschleunigungssensoren besitzt, um die Beschleunigungen in alle drei Raumdimensionen zu bestimmen. Diese Beschleunigungssensoren messen die Graviationskräfte. In Ruhelage wirkt auf jeden Körper eine Kraft von $1g \sim 9,80665 \frac{m}{s^2}$ (der Schwerkraft) in Richtung des Erdmittelpunktes. Damit lassen sich drei Freiheitsgrade der Wiimote bestimmen die entlang der Achsen x, y, z durch die Wiimote verlaufen, wie in Abb. 2.2 zu sehen ist. In diese Richtungen sind auch die Beschleunigungssensoren ausgerichtet. Die Beschleunigungssensoren der Wiimote sind mit 100Hz getaktet und lesen somit im 10 Millisekundentakt die Beschleunigungen der Wiimote im dreidimensionalen Raum aus.
Die Beschleunigungssensoren und auch der generelle Aufbau der Wiimote ist detailliert im Projektbericht [Hei09] Kapitel 2 beschrieben.

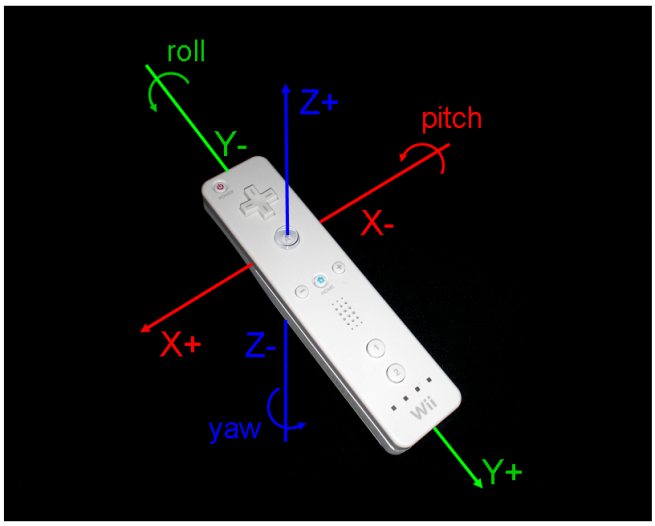

Abb. 2.2.: *Wii Orientierung*

Tool Zum Auslesen der Beschleunigungsdaten ist die Wiimote über Bluetooth mit einem Rechner verbunden (Details siehe Projektbericht [Hei09]). Auf diesem Rechner läuft eine selbst programmierte javabasierte Anwendung, deren grundlegende Funktionen bereits während der Projektarbeit 2009 implementiert wurden [4]. Diese Anwendung baut auf dem WiiGee-Framework

[4] Im Rahmen der Masterthesis wurden der zweite Aufzeichnungsmodus, eine nochmals überarbeitete Datenstruktur, ein zusätzliches Wiimote(Device)-Objekt mit deaktivierter Vorfilterung, eine (Re-)Kalibrierung der

von Poppinga [Pop07] auf[5]. Dieses Framework ermöglicht die Verwaltung der Verbindung zur Wiimote über eine Bluetooth-Schnittstelle mit dem Rechner. Hierüber lassen sich u.a. die Beschleunigungsdaten auslesen, aufzeichen, speichern und, mittels der dort implementierten Gestenerkennung über ein Hidden Markov Modell, wiedererkennen.

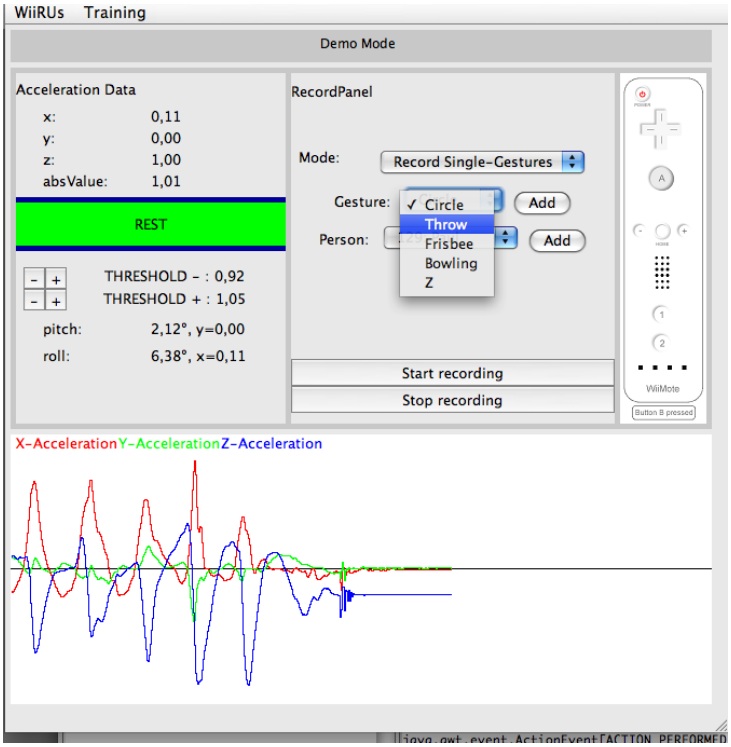

Abb. 2.3.: *GUI des entwickelten WiiTools zur Gestenaufzeichnung*

Das entwickelte WiiTool besitzt ein grafisches Interface (siehe Abb. 2.3), über das die Verbindung zur Wiimote verwaltet und eine Aufzeichnung der kontinuierlich ausgelesenen Wiimote-Beschleunigungsdaten initiiert werden kann. Die aktuellen Beschleunigungswerte werden ausgegeben und ihr Verlauf in einem Graphen dargestellt. Zusätzlich werden die Winkel *pitch* und *roll* zur Lagebestimmung der Wiimote berechnet. Unter Verwendung des Beschleunigungsbetrags ist ein Ruhelagedetektor implementiert, der anzeigt, ob sich die Wiimote gerade in Bewegung oder Ruhe befindet. Die Grenzen dieses Detektors, der die Ruhelage anhand von Schwellwerten um den Beschleunigungsbetrag herum misst, sind interaktiv veränderbar. Vor der Aufzeichnung von Daten lassen sich durch Nutzereingabe der Aufzeichnungsmodus, die Geste die aufgezeichnet werden soll und die Person definieren. Durch *start-* und *stop-recording* wird eine Aufzeichnungssequenz begonnen oder beendet. Die Daten werden in einer Textdatei gespeichert.

Zur Aufzeichnung und Speicherung der Beschleunigungsdaten wurde ein erweitertes Datenmodell entworfen, um z.B. auch Metadaten über den Probanden oder die jeweilige Versuchsdurchführung zu erfassen. Die eigentlichen Beschleunigungsvektoren, die aus einem Tripel der Beschleunigungen in die x-, y-, z-Koordinatenrichtungen bestehen, werden um einen Zeitstempel ergänzt. Die einzelnen Gesten werden bei der Aufzeichnung mit einer PatternID versehen, über die später die jeweilige Gestenklasse, aber auch die zusätzlichen Metadaten über den Probanden zugeordnet werden. Ein Ausschnitt der aufgezeichneten Daten mit der Java-Anwendung ist in 2.4 darge-

Daten und das Testmodul integriert.

[5]Verwendet werden die neuesten Libraries wiigee-lib und wiigee-plugin-wiimote, Stand: Nov, 2009.

stellt. Eine andere Datenstruktur war nicht nur notwendig, um die Daten zur Klassifizierung

```
1PatternID; GestureID; TimeStamp; x_acc; y_acc; z_acc; r; PersonID; StreamNo; TimeIndex
2720;1;9.751;-0.7407;-0.3704;0.2692;0.8708;0;0;1
3720;1;19.358;-0.5926;-0.4074;0.2308;0.7552;0;0;2
4720;1;29.239;-0.5926;-0.3703;0.2308;0.7359;0;0;3
5720;1;39.38;-0.7037;-0.3333;0.1154;0.7872;0;0;4
6720;1;49.386;-0.7037;-0.3704;0.0;0.7952;0;0;5
7    ... ... ...
```

Abb. 2.4.: *Beispielausschnitt der aufgezeichnete Gestendaten mit dem WiiTool*

mit der SFA zu verwenden, sondern auch gerade für die Segmentierung. Das WiiTool ermöglicht die Aufzeichnung in zwei Modi. Erstens zur reinen Klassifizierung, bei der Gestenstart und -ende über das Drücken bzw. Loslassen des A-Buttons der Wiimote gekennzeichnet und nur diese Teile aufgezeichnet werden. Zweitens, einen Segmentierungsmodus zur Aufzeichnung von kontinuierlichen Datenströmen, in denen eine Geste mehrmals hintereinander ausgeführt werden kann und die Nicht-Gesten-Bereiche dazwischen mit erfasst werden. Die Gesten werden dabei ebenfalls über den gedrückten Wiimote-Button markiert. Diese Datenströme mit fortlaufendem Zeitstempel werden zur Untersuchung der Segmentierung benötigt.

In den neuesten WiiGee-Libraries werden die Daten der Wiimote bei der Aufzeichnung vorgefiltert (in der alten WiiGee-Lib, die für die Projektarbeit verwendet wurde, waren die Daten ungefiltert). Diese Filter (Idle State und Directorial Equivalence Filter) werden direkt bei Erzeugung des Wiimote(Device)-Objektes an dieses gebunden. Um die ungefilterten Rohdaten für die Klassifizierungstests aufzuzeichnen, wurde diese HMM-Vorfilterung deaktiviert, indem ein zusätzliches Wiimote(Device)-Objekt ohne Filter implementiert wurde.

Zu Vergleichszwecken mit der SFA wird das im WiiGee bereits implementierte HMM verwendet. Hierzu wurde noch ein zusätzliches Testmodul entworfen, das die aufgezeichneten Rohdaten wieder einliest und die gleichen Tests durchführt, die auch für die Untersuchung mit der SFA zur Gestenerkennung verwendet wurden. Die gespeicherten Daten werden in das temporäre Datenmodell geladen, entsprechend dem gewählten Testverfahren (es wurden die in 2.4 beschriebenen drei Testverfahren implementiert) aufsplittet in Trainings- und Testset. Diese aufgesplitteten Daten aus dem neuen Datenmodell werden wieder in die ursprünglich im WiiGee-Framework verwendete Datenstruktur zurückgewandelt, indem die einzelnen Beschleunigungsdaten sowohl für das Training als auch für den Test wieder als separate Beschleunigungsevents dem System übergeben werden, so als würden sie gerade mit der Wiimote aufgezeichnet. Hierzu musste eine dritte Device-Klasse implementiert werden, mit der eine verbundene Wiimote simuliert wird. Das beinhaltet ein simuliertes Drücken des A-Buttons vor Beginn der Aufzeichnung, das Loslassen dieses Buttons am Ende der jeweiligen Geste und das Drücken des Home-Buttons zum Abschluss der Aufzeichnungsreihe für eine Geste. Die gewählten Trainingsgesten werden so nacheinander dem HMM antrainiert. Analog dazu erfolgt der anschließende Test ebenfalls mit dem simulierenden Device. Hiermit lässt sich dann das im WiiGee implementierte HMM trainieren und testen.

Verwendete Gesten

In dieser Arbeit wurden sowohl zur Klassifikation als auch zur Segmentierung folgende fünf Gesten verwendet:

Abb. 2.5.: *Verwendete Gesten: circle, throw, frisbee, bowling und z*

- *circle*: ein Kreis in der Luft gezeichnet
- *throw*: ein über Kopf Wurf (Schlagball-Weitwurf)
- *frisbee*: ein horizontaler Frisbee-Wurf (Drehbewegung aus dem Handgelenk)
- *bowling*: Schwung der Arms neben dem Körper mit Abwurf einer gedachten Bowlingkugel
- *z*: der Buchstabe "Z" in der Luft geschrieben

Eine ausführlichere Beschreibung der Gestencharakteristika ist ebenfalls im Projektbericht 2009 [Hei09] zu finden.

Zur personenübergreifenden Klassifizierung wurden die fünf ausgewählten Gesten von 10 Probanden aufgezeichnet, von denen jeder jede Geste mindestens 10x nacheinander ausführte. Zusätzliche Merkmale der Probanden wie Alter, Geschlecht, Rechtshänder/Linkshänder wurden ebenfalls erfasst, so dass eine spätere Unterscheidung anhand dieser Merkmale möglich ist. Die Gesten wurden unangeleitet aufgezeichnet, das heißt es war den Probanden freigestellt, wann für sie eine Geste beginnt oder endet und ob sie z.B. den Kreis im oder gegen den Uhrzeigersinn ausführen oder wo für sie eine Kreisgeste beginnt.

Für den Teil der Segmentierung wurden die fünf gewählten Gesten von einem Probanden in jeweils einem Datenstrom im zuvor beschriebenen Segmentierungsmodus aufgezeichnet. Jede dieser Aufzeichnungen hat eine Länge von mindestens 30 Sekunden. Die Anzahl der Gesten (und der dazwischenliegenden Ruhelagen), die in dieser Zeit ausgeführt wurden, variiert allerdings von Geste zu Geste.

Ein detaillierter Überblick über die jeweils verwendeten Daten ist in den Kapiteln 2.3.2 (für die Klassifizierung) und 3.2.1 (für die Segmentierung) zu finden.

2.3.2. Verwendete Daten

Für die Klassifizierung ist ein Set von 716 Trainingsgesten mit insgesamt 5 Gestenklassen von 10 Probanden aufgezeichnet worden. Dabei sind neun Gestensets von Rechtshändern und eines von einem Linkshänder. Eine Unterscheidung anhand von Alter und Geschlecht war für die bisherigen Versuche nicht notwendig. Ein Überblick über die aufgezeichneten Gestensets ist in Tabelle 2.1 dargestellt:

Tab. 2.1.: *Überblick über die verwendeten Gestendaten zur Klassifizierung*

	Person Metadata				Gestures					
#	personID	mf	hand	age	circle	throw	frisbee	bowling	z	sum
1	121	male	right	19	23	11	12	12	11	69
2	122	female	right	26	20	20	10	14	10	74
3	123	male	right	27	11	11	11	11	11	55
4	124	female	right	29	12	11	11	11	11	56
5	125	male	right	25	19	11	12	10	13	65
6	126	male	right	25	11	11	11	10	10	53
7	127	male	right	25	10	18	10	18	11	67
8	128	male	right		19	12	19	10	10	70
9	129	male	left		10	16	10	10	10	56
10	130	female	right	28	31	28	31	31	30	151
	Sum Pattern				166	149	137	137	127	716

Ein Überblick über die aufgezeichneten und für die Klassifizierung verwendeten Gestendaten anhand ihrer Durchführungszeit ist in Tabelle 2.2 dargestellt. Die minimale und maximale Ausführungsdauer einer Geste ist jeweils rot markiert. Über alle Gestendaten betrachtet variiert die Dauer für die Durchführung der einzelnen Gesten wie folgt:

- *circle*: 0,765s - 4,905s, im Mittel: 1,560s
- *throw*: 0,407s - 2,297s, im Mittel: 1,052s
- *frisbee*: 0,367s - 2,027s, im Mittel: 0,854s
- *bowling*: 0,521s - 2,291s, im Mittel: 1,233s
- *z*: 0,971s - 4,272s, im Mittel: 1,798s

Es lässt sich feststellen, dass für die Gesten *circle* und *z* mit im Durchschnitt 1,560s und 1,798s mehr Ausführungszeit benötigt wird als für die *throw*- und *bowling*-Gesten. Die kürzeste Ausführungsdauer hat durchschnittlich die *frisbee*-Geste mit 0,854s.
In einigen Fällen sind die Standardabweichungen bei Gesten ein und derselben Person relativ hoch, wie z.B. die *circle*-Geste der Person 122 oder die *z*-Geste der Person 128. Das bedeutet, dass innerhalb des Gestensets einer Person diese Geste bereits bewußt oder unbewußt sehr unterschiedlich ausgeführt wurde.

Tab. 2.2.: *Überblick Gestendauer pro Geste und pro Person im Durchschnitt (in Sekunden)*

	circle				**throw**			
personID	min	max	avg ± std	median	min	max	avg ± std	median
121	1,254	2,141	1,690 ± 0,240	1,672	0,407	0,720	0,532 ± 0,086	0,548
122	1,446	4,905	2,106 ± 0,763	2,022	0,541	1,759	1,039 ± 0,389	0,881
123	1,160	1,395	1,292 ± 0,059	1,288	0,534	0,878	0,671 ± 0,101	0,644
124	1,059	1,526	1,275 ± 0,138	1,266	0,444	0,703	0,576 ± 0,069	0,583
125	0,975	1,810	1,548 ± 0,174	1,582	1,610	2,123	1,762 ± 0,144	1,720
126	1,539	1,739	1,649 ± 0,069	1,650	0,915	1,382	1,144 ± 0,125	1,145
127	1,067	1,617	1,392 ± 0,174	1,402	0,559	1,142	0,827 ± 0,155	0,801
128	1,618	2,210	1,828 ± 0,139	1,819	1,819	2,297	1,998 ± 0,147	1,933
129	1,159	2,396	2,109 ± 0,336	2,216	1,083	1,614	1,352 ± 0,169	1,332
130	0,765	1,376	1,005 ± 0,116	0,972	0,633	1,035	0,856 ± 0,122	0,904
all	0,765	4,905	**1,560** ± 0,481	1,569	0,407	2,297	**1,052** ± 0,464	0,937
	frisbee				**bowling**			
personID	min	max	avg ± std	median	min	max	avg ± std	median
121	0,518	0,890	0,714 ± 0,114	0,712	0,843	1,995	1,358 ± 0,340	1,363
122	0,868	1,148	0,977 ± 0,077	0,974	1,235	1,846	1,507 ± 0,173	1,477
123	0,505	0,702	0,587 ± 0,054	0,580	0,527	0,800	0,675 ± 0,080	0,674
124	0,431	0,775	0,629 ± 0,100	0,634	0,530	1,106	0,812 ± 0,176	0,788
125	1,317	2,027	1,811 ± 0,185	1,813	1,668	2,291	2,024 ± 0,185	1,982
126	0,849	1,163	1,029 ± 0,088	1,024	0,855	1,125	1,022 ± 0,080	1,010
127	0,630	1,294	0,955 ± 0,190	0,925	1,024	1,800	1,352 ± 0,174	1,316
128	0,437	1,208	0,792 ± 0,228	0,763	1,359	1,787	1,582 ± 0,136	1,617
129	0,901	1,411	1,102 ± 0,148	1,068	1,080	1,570	1,281 ± 0,157	1,246
130	0,367	0,744	0,536 ± 0,074	0,524	0,521	1,369	1,023 ± 0,196	1,029
all	0,367	2,027	**0,854** ± 0,378	0,744	0,521	2,291	**1,233** ± 0,391	1,230

	z			
personID	min	max	avg ± std	median
121	2,071	2,755	2,388 ± 0,204	2,350
122	1,973	2,334	2,203 ± 0,106	2,236
123	1,327	1,568	1,437 ± 0,078	1,429
124	1,335	1,680	1,521 ± 0,090	1,509
125	1,601	2,324	1,902 ± 0,172	1,852
126	1,108	1,781	1,338 ± 0,215	1,231
127	1,526	1,776	1,656 ± 0,086	1,695
128	2,101	4,272	2,509 ± 0,607	2,356
129	2,270	2,701	2,487 ± 0,123	2,461
130	0,971	1,732	1,376 ± 0,156	1,371
all	0,971	4,272	**1,798** ± 0,499	1,627

2.3.3. Vorverarbeitung

Um die Daten für die Verwendung mit der SFA aufzubereiten und zu normieren, wurden vier Vorverarbeitungsschritte vorgenommen (Interpolation, Längen- und Amplitudennormierung und Konkatenieren der Zeitreihen). Die Vorverarbeitung wurde in Matlab implementiert.

Zuerst wird jede x-, y-, z-Zeitreihe eines Pattern linear interpoliert. Dies ist notwendig, da die Daten nicht in äquidistanten Abstanden ausgelesen wurden.

Da die Pattern keine einheitliche Länge besitzen, wird im zweiten Schritt eine Längennormierung (siehe Abb. 2.6) durchgeführt, so dass alle Pattern hinterher eine einheitliche Länge besitzen bzw. aus n Abtastpunkten bestehen (hier: n= 30). Aus den zuvor interpolierten Zeitreihen werden dazu n-Werte an den Zeitpunkten $t_n = n*\Delta t + \frac{\Delta t}{2}$, mit $0 \leq k < n$ ausgelesen. Anschließend werden diese geglättet, d.h. eigentlich werden für jeden Zeitpunkt t die N (z.B. N = 10) nächsten Werte mit betrachtet. Ihr Mittelwert wird als Repräsentant der Werte zum Zeitpunkt t verwendet.

$$x_{new}(k) = \frac{1}{N} \sum_{j=k*N}^{(k+1)*N-1} x_{interp}(t_j) \tag{2.7}$$

, mit

$$t_j = j * \Delta t_G + \frac{\Delta t_G}{2} \tag{2.8}$$

, und

$$\Delta t_G = \frac{t_{max}}{N * dim} \tag{2.9}$$

N ist dabei die Anzahl der Werte über deren Mittelwert geglättet wird und *dim* die neue Dimension der einzelnen Zeitreihen.

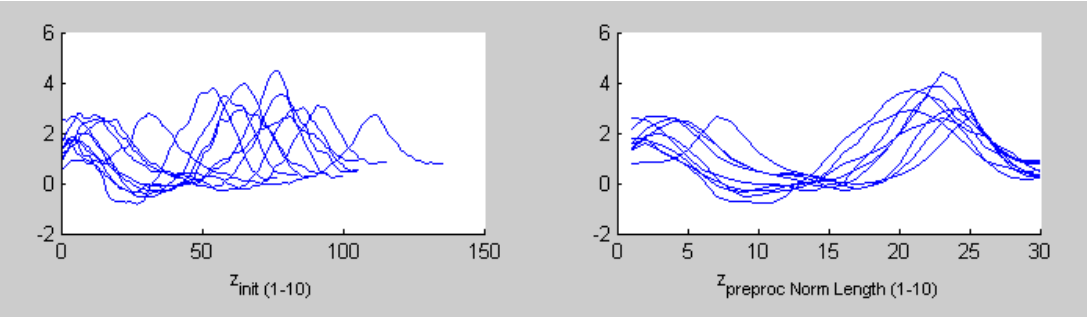

Abb. 2.6.: *Längennormierung (Bsp.Kreis z-Beschleunigung) vorher - nachher*

Um die Beschleunigungsdaten auch von ihrer Beschleunigungsstärke zu vereinheitlichen, wird im dritten Schritt der Vorverarbeitung zur Normierung der Amplitude die Standardabweichung verwendet. Die Standardabweichung ist der Durchschnitt der Abweichung aller Werte von ihrem Mittelwert. Dividiert man die komplette Zeitreihe durch ihre Standardabweichung, so normiert man damit die Amplitude des Signals, so dass die Standardabweichung des neuen Signals = 1 ist. Die Auswirkungen der Amplitudennormierung auf die SFA ist zusätzlich zu der grafischen Darstellung in Abb. 2.7 auch in der Gegenüberstellung in Tabelle 2.8 ersichtlich.

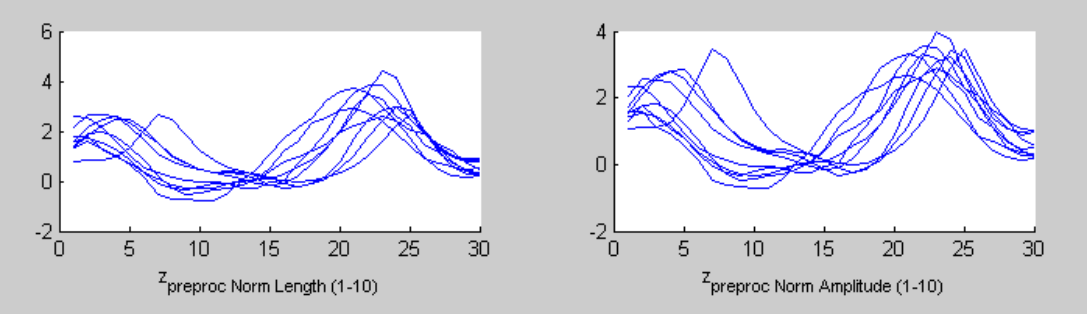

Abb. 2.7.: *Amplitudennormierung (Bsp.Kreis z-Beschleunigung) vorher/nachher*

Als Eingabedaten für die SFA werden die konkatenierten Zeitreihen der x-, y-, z-Beschleunigungsdaten verwendet. Diese werden zusätzlich um einen Wert, nämlich die Dauer des jeweiligen Pattern, ergänzt. Die Ausgabedaten setzen sich dann wie folgt zusammen:

$$\vec{p} = [acc_x(1), ..., acc_x(n), acc_y(1), ..., acc_y(n), acc_z(1), ..., acc_z(n), max(TimeStamp)/1000] \tag{2.10}$$

Wie Abb. 2.8 dargestellt, können diese Daten anschließend zur Klassifizierung, z.B. mit der SFA, weiter verwendet werden.
Zu welcher Klasse das jeweilige Pattern gehört, wird für die spätere Zuordnung der Trainings- und Testdaten zusätzlich abgespeichert.

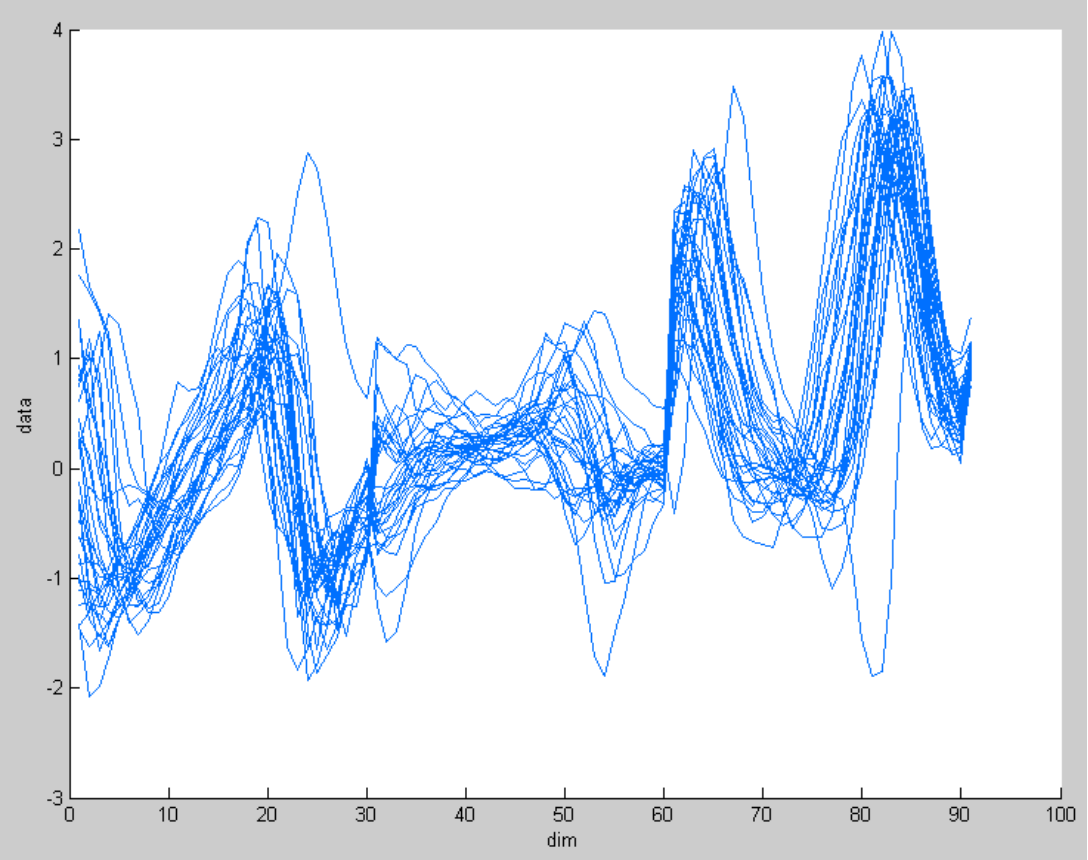

Abb. 2.8.: *Zeitreihe der Kreis-Geste nach kompletter Vorverarbeitung*

Detailbeschreibung der Vorverarbeitung (siehe Projektbericht).

2.4. Klassifizierungstests

Es sollte untersucht werden, ob mit der SFA eine Klassifizierung von Gesten möglich ist. Die Erhebung von Daten war bereits darauf ausgelegt, nicht nur die Gesten einzelner Personen, sondern auch personenübergreifend zu klassifizieren. Anfänglich wurde die Klassifizierung mit der SFA personenabhängig (Test III) und über die gesamte Datenmenge (Test IV) getestet. Bei personenabhängigen Tests kam es zu Schwierigkeiten, da hier für eine Klassifizierung mit SFA teilweise nicht ausreichend Gestendaten der einzelnen Personen vorhanden waren, wie sich im weiteren Verlauf herausstellte. An dieser Stelle kam ein weiterer Klassifizierungstest hinzu, für den ausreichend große Trainingssets für die SFA vorhanden waren, nämlich der Test mit den Gesten einer neuen Person (Test V).

Bei weiteren Tests mit der SFA hat sich herausgestellt, dass die SFA beim Variieren der Reduktionsdimension (das Verfahren der SFA beinhaltet eine Reduktion der Eingangsdaten mittels PCA) und der Größe der Trainingsdatenmenge zu sehr unterschiedlichen Ergebnissen kommt. Bei weiteren Untersuchungen hat sich gezeigt, dass so genannte "rank deficiency"-Probleme auftreten, die zu hohen Fehlerraten führen, wenn zu wenig Trainingsdaten im Verhältnis zur Reduktionsdimension verwendet werden. Diese Problematik wird deswegen zunächst in 'Test I - Auswirkung der Reduktion durch die PCA auf das Ergebnis' (Abschnitt 2.4.2) aufgegriffen und der Zusammenhang näher beschrieben.

Aufgrund dieser Problematik und der daraus resultierenden Tatsache, dass für einige Tests mit der SFA zu wenig Trainingsdaten vorhanden waren, wurde darüber nachgedacht, die Originaltrainingsdaten um zusätzliche künstlich erzeugte Trainingsdaten zu ergänzen. Hierzu wurden mehrere unterschiedliche Möglichkeiten für ein so genanntes *Parametric Bootstrap* untersucht. Die Untersuchung möglicher Verfahren und eine Gegenüberstellung sind in 'Test II - Parametric Bootstrap' (Abschnitt 2.4.3) zu finden.

In den darauf folgenden Abschnitten werden dann die eigentlichen Tests zur personenabhängigen (Test III) und personenübergreifenden (Test IV) Klassifizierung beschrieben. Teilweise war für diese Tests das zuvor beschriebene *Parametric Bootstrap* notwendig, so dass die Ergebnisse aus der vorherigen Untersuchung in diese Tests mit einflossen. Ebenfalls wurde untersucht, ob eine Klassifizierung mit der SFA möglich ist, wenn als Testdaten die Gesten einer oder mehrerer neuer Personen verwendet werden, deren Daten nicht zuvor zum Training des Modells genutzt wurden (Test V).

Im Ansatz von Mäntyjärvi et al. [MKKK04] wurde gezeigt, dass es mit ihrem HMM-basierten Ansatz zur Gestenerkennung möglich ist, mit nur zwei Originaltrainingspattern gute Ergebnisse (Fehlerraten von nur $\sim 3\%$) zu erreichen, wenn dort ebenfalls PB angewendet wird. Lernmodelle mit wenigen Originaldaten trainieren zu können, kann gerade in realen Umgebungen interessant sein. Deswegen soll in Test VI eine personenabhängige Klassifizierung mit der SFA unter Verwendung von Trainingssets mit einer minimalen Anzahl an Originaldaten untersucht werden.

2.4.1. Durchführung der Untersuchungen

Die Tests III-V wurden sowohl mit SFA (und dem Gaussklassifizierer) als auch mit dem HMM-Verfahren mittels Cross-Validation (CV) durchgeführt. Zwei unterschiedliche Arten der CV wurden hierzu eingesetzt:

- Die **10fache Cross-Validation** ist ein Verfahren bei dem die gesamte Datenmenge in zehn zufällig gewählte etwa gleichgroße Teile aufgeteilt wird. 90% (neun Teile) werden

zum Trainieren des Modells und der 10. Teil als Testmenge verwendet. Diese Prüfung wird zehnmal durchgeführt, jeweils mit einem anderen 10. Teil als Testmenge.

- Für Test V wurde leave-one-out Cross-Validation verwendet, in dem zum Training die Daten von 9 Personen verwendet wurden und das Testset dann aus den Daten der fehlenden 10. Person besteht. Dies wird für jede Person einmal durchgeführt.

Die zusammengefassten Ergebnisse der CV sind jeweils in einer Konfusionsmatrix dargestellt. Eine Konfusionsmatrix dient zur Beurteilung eines Klassifikators, indem in einer quadratischen Tabelle die Häufigkeiten des Auftretens für alle möglichen Kombinationen von ermittelter Klasse und tatsächlicher Klasse eingetragen werden. Wie die jeweilige CV in den einzelnen Fällen durchgeführt wurde, ist in den entsprechenden Abschnitten III-V näher beschrieben.

In den durchgeführten Klassifizierungstests wird die SFA so verwendet, wie sie bereits von Berkes zur Mustererkennung angewandt wurde und in Kapitel 2.2 beschrieben ist. Alle Klassifizierungstests werden mit einer SFA 2.Grades durchgeführt. Verwendet wird für die Tests mit der SFA das Matlab-TK V2.6 [Kon09]. Die SFA-Testskripte sind so gestaltet, dass parallel zu den SFA-Tests die gleichen Daten mit einen Gaussklassifizierer getestet werden (der Gaussklassifizierer wurde ebenfalls in Kapitel 2.2 beschrieben).
Für die Klassifizierungstests wurde jeweils ein Teil der verwendeten Daten als Trainingsset verwendet, um zunächst das jeweilige Modell zu trainieren. Anschließend wurde in allen Fällen jeweils der andere Teil der Daten (der nicht zum Training verwendet wurde) zum Testen des Modells verwendet.
Die Fehlerraten der Tests mit SFA- und Gaussklassifizierer werden aber sowohl für Test- als auch für Trainingsdaten zurückgegeben. Sie werden ermittelt durch:

$$\text{Fehlerrate} = \frac{\text{Anzahl falsch klassifizierte Pattern}}{\text{Gesamtanzahl Pattern}} \tag{2.11}$$

Für alle in Matlab implementierten SFA-Tests wurden als Pattern P, die komplett vorverarbeiteten 91-dimensionalen Eingangsvektoren verwendet.

$$P = \{x_{acc_1}, ..., x_{acc_n}, y_{acc_1}, ..., y_{acc_n}, z_{acc_1}, ..., z_{acc_n}, len\}, n = 30 \tag{2.12}$$

Für die Klassifizierungstests II-VI mit der SFA wurde eine Reduktionsdimension (*pprange*) von 12 gewählt, d.h. dass im Reduktionsschritt der SFA der Eingangsvektor mittels PCA auf 12 Dimensionen reduziert wird. Die gewählte Dimension ist ein Kompromiss von Laufzeit und SFA-Ergebnis.

In den Klassifizierungstests mit dem HMM-Verfahren wurden die vorliegenden Gestendaten so verwendet, wie sie zuvor mit der Java-Anwendung, basierend auf dem WiiGee-Framework, aufgezeichnet wurden, also ohne (SFA-)Vorverarbeitung. Wie bereits in Abschnitt 2.3.1 beschrieben wurde für Klassifizierungstests mit dem HMM-Verfahren ein Testmodul in der Java-Anwendung implementiert. Dieses Testmodul verwendet die Gestenerkennungsfunktionen, die im WiiGee-Framework enthalten sind. Diese Gestenerkennungsfunktionen basieren auf dem HMM-Verfahren. Im Testmodul werden die Gestenerkennungsfunktionen genauso verwendet wie sie in den jeweiligen Libraries und den dazugehörigen Testanwendungen verfügbar sind.

Für die Untersuchung von 'Test V - Erkennung der Gesten einer neuen Person' (Abschnitt 2.4.6) lagen aus anderen Arbeiten Koch et al. [KKH10] Ergebnisse zur Klassifizierung mit Random Forest (RF) vor. Bei diesen Untersuchungen wurden die gleichen Daten wie für die Klassifizierung mit SFA verwendet und deswegen können auch diese Ergebnisse zum Vergleich mit einem anderen gängigen Klassifizierungsverfahren herangezogen werden. Random Forest ist ein von Breiman

[Bre01] vorgestellter Klassifizierer, der auf einer Menge von Entscheidungsbäumen basiert, die während des Lernprozesses wachsen.

2.4.2. Test I - Auswirkung der Reduktion durch die PCA auf das Ergebnis

Bei ersten Klassifizierungstests mit der SFA hat sich gezeigt, dass die Anzahl der verwendeten Trainingsdaten im Verhältnis zur Reduktionsdimension der PCA steht, die in der SFA verwendet wird. Laut Berkes [Ber05] wird die Genauigkeit der SFA mit steigender Eingangsdimension höher. Dieser Zusammenhang und die daraus resultierenden Erkenntnisse sollen anhand eines Testlaufs mit steigendem Reduktionsparameter *pprange* = [3..80] bei gleichbleibender Größe des Trainingssets $numtrn = 644$ gezeigt werden.
Die gemittelten Ergebnisse dieser 10fach CV-Tests sind in Tabelle 2.3 für SFA- und Gaussklassifizierer dargestellt. Die Reduktionsdimension entspricht dem Reduktionsparameter *pprange*. Ab der Reduktionsdimension 50 konnten aufgrund von Speicherproblemen nur noch einfache Ergebnisse ermittelt werden. Die Laufzeit steigt im Zusammenhang mit der Expansionsdimension D_{xp} stark an. Die Expansionsdimension D_{xp} entspricht der Länge des expandierten Vektors, der im Expansionsschritt der SFA aus den Monomen des reduzierten Eingangsvektors gebildet wird. Dargestellt ist hier die durchschnittliche Laufzeit für eine einmalige Durchführung der SFA.

Bei Reduktionen auf $pprange < 25$ zeigt die Klassifizierung mit SFA gute Ergebnisse, während bei höheren Dimensionen die Ergebnisse ab $D_{xp} \approx 500$ rapide schlechter werden. Die besten Ergebnisse auf gegebenen Daten liefern 18 Eingangsdimensionen. Ist die Expansionsdimension D_{xp} annähernd so groß wie die Anzahl der Trainingsdaten *numtrn* oder größer, liegen die Fehlerraten bei $\sim 80\%$ auf den Testdaten. Werden zum Testen ebenfalls die Trainingsdaten verwendet, so sind diese sehr gut mit annähernd 0% Fehlerrate auf dem Trainingsset klassifiziert worden. Dies wird als Overfitting bezeichnet. Das lernende System ist dabei für die trainierten Daten so überspezialisiert, dass andere Daten nur schlecht erkannt werden. Dieses Problem ist auch aus dem Bereich Neuronaler Netze bekannt.

Der Grund für die massiv schlechten Ergebnisse auf den Testdaten wenn die Reduktionsdimension und damit auch Expansionsdimension größer wird, liegt in dem Verfahren der SFA.
Wie in Abschnitt 2.2 beschrieben, werden die Eingangsvektoren im ersten Schritt der SFA mit Hilfe der PCA in ihrer Dimension reduziert. Im nächsten SFA-Schritt werden die so reduzierten Eingangsvektoren expandiert und eine Kovarianzmatrix der Größe D_{xp}xD_{xp} erstellt. Ist die Expansionsdimensionen D_{xp} größer als die Anzahl der Trainingsdaten *numtrn* abzüglich der Anzahl der unterschiedlichen Klassen K, treten so genannte "rank deficiency"-Probleme[6] auf. Dieses Problem ist erkennbar an den schlechten Ergebnissen mit $\sim 80\%$ Fehlerrate auf den Testdaten. Der Zusammenhang, dass als Bedingung für ein gutes Trainingsergebnis

$$numtrn \geq D_{xp} + K \tag{2.13}$$

gelten muss, ist in Koch et al. [KKH10] näher beschrieben. Aus diesem Grund ist für die Klassifizierungstests mit SFA eine Mindestanzahl an Trainingsdaten notwendig, wenn die Reduktionsdimension *pprange* festgelegt wird. In einigen Fällen liegen allerdings nicht ausreichend Trainingsdaten vor, so dass daraus die Überlegung entstand, die Originaltrainingsdaten durch künstlich erzeugte Trainingsdaten (Parametric Bootstrap) zu ergänzen.

[6] "rank deficiencies" treten dann auf, wenn eine Matrix keinen vollen Rang hat und dadurch das lineare Gleichungssystem unterbestimmt ist.

Tab. 2.3.: *SFA CV-Test mit steigender PCA-Reduktionsdimension pprange 3-80 und gleichbleibender Größe des Trainingssets*

pprange	SFA	Gauss	numtrn	D_{xp}	time(1xSFA)
3	(22.486 ± 0.838)%	(24.930 ± 0.140)%	644	9	1.0064 sec
4	(16.830 ± 0.140)%	(21.788 ± 0.559)%	644	14	1.3128 sec
5	(13.408 ± 0.279)%	(20.391 ± 0.838)%	644	20	1.6467 sec
6	(8.659 ± 0.000)%	(14.735 ± 1.257)%	644	27	2.0815 sec
7	(8.101 ± 0.000)%	(14.804 ± 0.279)%	644	35	2.5228 sec
8	(6.355 ± 0.419)%	(14.525 ± 0.559)%	644	44	3.0138 sec
9	(5.307 ± 0.279)%	(14.525 ± 0.559)%	644	54	3.5663 sec
10	(4.539 ± 0.419)%	(14.525 ± 0.838)%	644	65	4.4288 sec
11	(3.003 ± 0.140)%	(13.617 ± 0.140)%	644	77	4.7959 sec
12	(3.003 ± 0.978)%	(14.385 ± 0.838)%	644	90	5.8074 sec
13	(2.723 ± 0.698)%	(14.036 ± 0.140)%	644	104	6.3716 sec
14	(2.654 ± 0.279)%	(13.757 ± 0.140)%	644	119	7.5686 sec
15	(2.235 ± 0.000)%	(13.547 ± 0.279)%	644	135	8.1095 sec
16	(2.165 ± 0.419)%	(13.547 ± 0.559)%	644	152	9.3680 sec
17	(2.444 ± 0.698)%	(11.802 ± 0.140)%	644	170	9.9385 sec
18	(2.025 ± 0.140)%	(11.173 ± 0.559)%	644	189	11.5491 sec
19	(2.235 ± 0.000)%	(11.313 ± 0.838)%	644	209	12.1092 sec
20	(2.514 ± 0.279)%	(12.291 ± 1.117)%	644	230	13.9103 sec
21	(2.165 ± 0.140)%	(11.662 ± 0.140)%	644	252	15.3066 sec
22	(2.584 ± 0.698)%	(11.732 ± 0.279)%	644	275	16.2255 sec
23	(2.793 ± 0.559)%	(12.360 ± 0.419)%	644	299	17.6155 sec
24	(2.444 ± 0.140)%	(12.151 ± 0.838)%	644	324	19.1214 sec
25	(2.863 ± 0.419)%	(13.198 ± 0.140)%	644	350	21.4223 sec
29	(5.796 ± 0.140)%	(12.570 ± 0.279)%	644	464	28.2001 sec
30	(8.520 ± 0.279)%	(12.221 ± 0.419)%	644	495	31.5966 sec
31	(11.453 ± 2.235)%	(11.872 ± 0.000)%	644	527	32.3516 sec
32	(23.673 ± 3.492)%	(11.662 ± 0.140)%	644	560	34.7245 sec
33	(40.852 ± 0.419)%	(11.592 ± 0.000)%	644	595	38.4569 sec
34	(72.556 ± 0.140)%	(11.662 ± 1.257)%	644	629	43.2067 sec
35	(78.911 ± 1.676)%	(11.732 ± 0.279)%	644	665	47.6397 sec
36	(81.145 ± 0.559)%	(13.059 ± 0.140)%	644	702	50.2269 sec
37	(79.050 ± 2.235)%	(12.151 ± 0.559)%	644	740	52.4116 sec
38	(79.050 ± 1.955)%	(13.268 ± 1.117)%	644	779	55.9347 sec
39	(81.075 ± 1.257)%	(12.849 ± 0.279)%	644	819	48.1159 sec
40	(80.447 ± 0.279)%	(13.338 ± 0.698)%	644	860	50.9482 sec
50	80.028%	16.061%	644	1325	116.4058 sec
60	83.659%	18.436%	644	1890	212.8819 sec
70	81.145%	19.693%	644	2555	374.4126 sec
80	79.890%	21.523%	644	3320	771.7789 sec

2.4.3. Test II - Parametric Bootstrap

Wie in Test I beschrieben, ist mit der Festlegung auf eine Reduktionsdimension eine davon abhängige Mindestanzahl an Trainingsdaten notwendig. Da nicht für alle Klassifikationstests ausreichend Trainingsdaten vorliegen, sollen die vorliegenden Originaltrainingsdaten durch künstlich erzeugte Daten ergänzt werden. Diese Erzeugung neuer künstlicher Daten aus einem Originaldatenset wird als Parametric Bootstrap (PB) bezeichnet [HTF01].

Für das PB wird eine zufällige Auswahl von *ncopies*[7] Pattern aus der Menge der Originaltrainingsdaten verwendet [8]. Diese Pattern werden minimal verändert und anschließend als neue Pattern zu den Originaltrainingsdaten hinzugefügt. Die Originale bleiben erhalten. Diese Veränderung der Originaldaten kann auf unterschiedliche Weise erfolgen, z.B. durch Verrauschen, Rotieren, Kürzen etc.

Um zu zeigen, welche PB-Verfahren für Klassifizierungstest mit der SFA geeignet sind, wurden folgende Verfahren untersucht:

- Verrauschen der Originaldaten
- Rotieren der Originaldaten
- Kürzen der Originaldaten
- Kombinationen von Kürzen und Rotieren der Originaldaten

Diese werden im Folgenden kurz beschrieben und anschließend einander gegenübergestellt.

PB durch Verrauschen der Originaldaten

Beim Verrauschen der Daten werden zufällige Werte aus der Normalverteilung um den Zentroiden des Beschleunigungsvektors verwendet.

PB durch Rotieren der Originaldaten

Für jedes zufällig gewählte Pattern P (2.12) wird eine eigene Rotationsmatrix berechnet, mit der dieses Pattern rotiert wird. Jedes Pattern besteht aus einer zeitlichen Abfolge von dreidimensionalen Vektoren $v_{orig}(t)\top = (x_{acc_t}, y_{acc_t}, z_{acc_t})$. Für die Rotationen in alle drei Koordinaten-Richtungen werden folgende drei Rotationsmatrizen (für Zeilenvektoren, v wird transponiert) verwendet

$$M_{rotX} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & \sin(\alpha) \\ 0 & -\sin(\alpha) & \cos(\alpha) \end{pmatrix}, M_{rotY} = \begin{pmatrix} \cos(\beta) & 0 & -\sin(\beta) \\ 0 & 1 & 0 \\ \sin(\beta) & 0 & \cos(\beta) \end{pmatrix}, M_{rotZ} = \begin{pmatrix} \cos(\gamma) & \sin(\gamma) & 0 \\ -\sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

die beliebig kombiniert werden können. Die Kombination gibt die Abfolge der nacheinander ausgeführten Rotationen an. Eine Vertauschung der Reihenfolge führt auch zu einer Änderung der Abfolge der Rotationen. Die Ergebnisse sind nicht identisch. Somit muss festgelegt werden, in welcher Reihenfolge die jeweiligen Vektoren rotiert werden. Gewählt wurde:

$$M_{rot} = M_{rotX} * M_{rotY} * M_{rotZ} \tag{2.14}$$

Pro Pattern werden zufällige Winkel jeweils für α, β und γ gewählt, die alle kleiner einem vorher definierten maximaler Rotationswinkel δ_{MAX} sind, $-\delta_{MAX} \leq \alpha, \beta, \gamma \leq \delta_{MAX}$. Mit diesen

[7] *ncopies* ist der in den Matlab-Skripten verwendete Parameter zur Bestimmung der Anzahl der Kopien.
[8] mit Zurücklegen, deswegen kann *ncopies* auch größer als die Anzahl der Originaltrainingsdaten sein.

Winkeln wird die zuvor beschriebene Rotationsmatrix berechnet. Die transponierten Beschleunigungsvektoren des gewählten Patterns werden nun einzeln mit der Rotationsmatrix multipliziert $v_{rot}(t)^\top = v_{orig}(t)^\top * M_{rot}$ und anschließend zu einem neuen rotierten Pattern wieder zusammen gesetzt.

Experimentelle Tests auf den Daten haben gezeigt, dass ein Winkel $\geq 35°$die besten Ergebnisse liefert. Deswegen wurde in allen in dieser Arbeit verwendeten Untersuchungen der Maximalwinkel $\delta_{MAX} = 35°$ gesetzt.

PB durch Kürzen der Originaldaten

Beim Erzeugen von neuen künstlichen Daten durch Kürzen werden die Originalpattern um m zufällige Werte, $m \leq 3$, vorne und hinten gekürzt und anschließend durch Interpolation wieder auf die vorherige Anzahl an Werten "gestreckt".

Vergleich der unterschiedlichen PB - Verfahren

Es sollte getestet werden, wie gut das jeweilige Verfahren zur Erzeugung künstlicher Pattern im Vergleich zu den anderen ist.

Für die Untersuchung wurden die Originaldaten der Person 122 verwendet und durch die zuvor beschrieben Verfahren mit zusätzlichen künstlichen Kopien erweitert. Die Ergebnisse wurden über 10fache CV in 10 Läufen ermittelt und sind detailliert in Tabelle 2.4 dargestellt. In der zugehörigen Grafik 2.9 soll das Verhältnis nochmals verdeutlicht werden.

Tab. 2.4.: *Vergleich Bootstrap über 0-600 PB-Pattern (Person 122)*

ncopies	CV SFA(original + copies)				CV Gauss	numtrn
	noise	rotation	shorted	shorted+rotation		
0	(80.946 ± 3.941)%	(81.486 ± 3.681)%	(79.189 ± 4.758)%	(81.486 ± 3.224)%	(7.838 ± 1.124)%	66
10	(81.216 ± 3.453)%	(81.486 ± 4.087)%	(80.405 ± 4.207)%	(78.514 ± 5.391)%	(6.351 ± 2.020)%	76
20	(78.514 ± 5.755)%	(80.135 ± 2.773)%	(80.541 ± 4.310)%	(77.973 ± 4.354)%	(6.351 ± 2.426)%	86
30	(59.054 ± 7.108)%	(61.351 ± 8.392)%	(78.784 ± 5.457)%	(58.514 ± 6.547)%	(6.622 ± 2.272)%	96
40	(18.649 ± 6.572)%	(20.405 ± 3.182)%	(49.730 ± 5.488)%	(17.027 ± 3.927)%	(7.027 ± 1.992)%	106
50	(9.459 ± 3.553)%	(12.297 ± 4.271)%	(33.243 ± 6.586)%	(11.216 ± 3.293)%	(7.027 ± 1.471)%	116
60	(5.000 ± 2.331)%	(9.054 ± 2.605)%	(22.027 ± 3.975)%	(6.486 ± 2.307)%	(6.622 ± 2.369)%	126
70	(4.189 ± 1.705)%	(7.297 ± 0.996)%	(14.324 ± 4.413)%	(6.351 ± 3.556)%	(6.216 ± 2.037)%	136
80	(3.649 ± 2.129)%	(6.757 ± 3.424)%	(11.216 ± 2.426)%	(7.162 ± 2.773)%	(6.351 ± 1.173)%	146
90	(2.297 ± 1.173)%	(5.811 ± 2.331)%	(10.946 ± 2.639)%	(4.595 ± 1.531)%	(6.892 ± 1.705)%	156
100	(2.568 ± 1.417)%	(5.676 ± 2.912)%	(7.838 ± 3.276)%	(4.865 ± 2.618)%	(6.757 ± 2.227)%	166
110	(2.568 ± 1.417)%	(4.459 ± 2.517)%	(6.757 ± 2.326)%	(4.730 ± 1.929)%	(7.027 ± 2.307)%	176
120	(2.703 ± 2.014)%	(3.919 ± 1.568)%	(7.973 ± 2.463)%	(2.973 ± 1.992)%	(5.541 ± 1.417)%	186
130	(2.027 ± 1.007)%	(4.324 ± 2.402)%	(4.730 ± 1.539)%	(3.649 ± 2.331)%	(6.757 ± 2.421)%	196
140	(1.757 ± 1.173)%	(4.595 ± 2.530)%	(3.378 ± 1.808)%	(3.514 ± 2.345)%	(6.622 ± 1.417)%	206
150	(2.568 ± 2.171)%	(3.919 ± 1.417)%	(5.405 ± 1.899)%	(3.243 ± 1.201)%	(5.946 ± 2.145)%	216
200	(1.622 ± 1.617)%	(3.784 ± 1.617)%	(3.649 ± 2.773)%	(3.108 ± 1.351)%	(7.027 ± 2.207)%	266
250	(1.216 ± 1.247)%	(4.189 ± 1.705)%	(3.514 ± 2.037)%	(1.892 ± 0.996)%	(6.081 ± 1.384)%	316
300	(2.297 ± 1.509)%	(3.919 ± 1.417)%	(3.243 ± 1.531)%	(2.568 ± 1.833)%	(7.027 ± 1.124)%	366
400	(1.892 ± 0.736)%	(3.649 ± 1.783)%	(2.703 ± 1.502)%	(1.351 ± 1.343)%	(7.162 ± 2.129)%	466
500	(2.162 ± 0.996)%	(2.973 ± 1.617)%	(1.892 ± 1.376)%	(1.351 ± 0.950)%	(7.027 ± 1.617)%	566
600	(2.432 ± 0.601)%	(3.514 ± 1.201)%	(2.027 ± 1.007)%	(2.703 ± 1.899)%	(6.622 ± 1.568)%	666

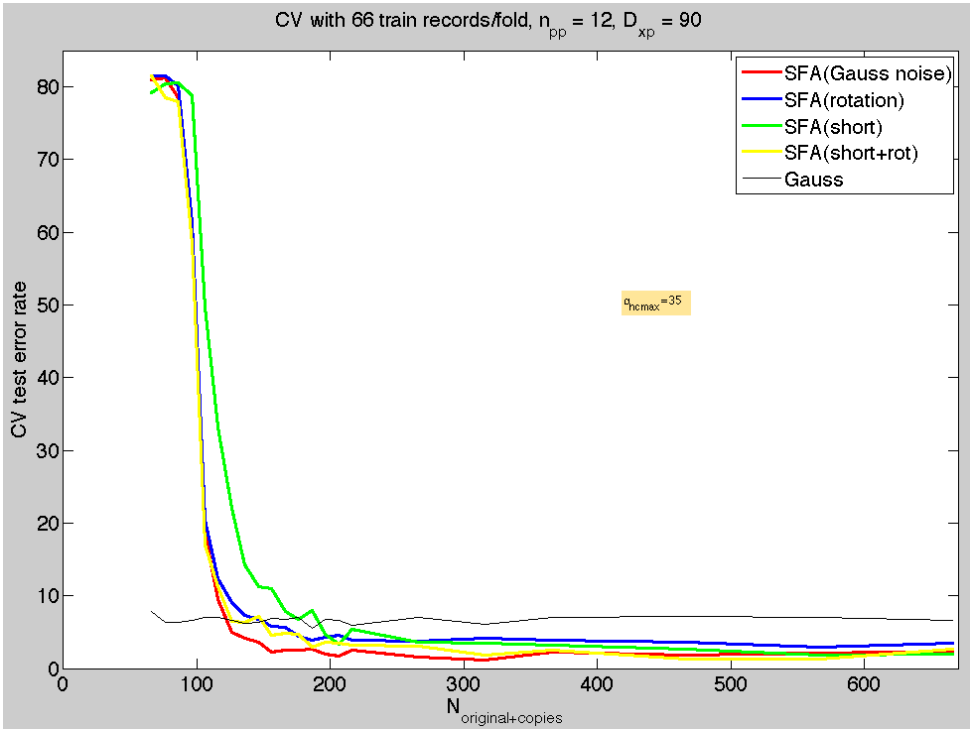

Abb. 2.9.: *Vergleich Bootstrap-Verfahren Rauschen, Drehen, Kürzen, Kürzen+Drehen über 0-600 Kopien + 66 Originaldaten der Person ID 122 angewandt mit SFA und exemplarisch einmal mit Gauss*

Grundsätzlich verbessern sich alle Ergebnisse, sobald die Anzahl von künstlichen Kopien *ncopies* und Originaldaten zusammen größer als 90 wird, das entspricht der Dimension des expandierten Eingangsvektors (Erklärung "rank deficency"-Problem siehe 2.4.2). In diesem Fall werden die Ergebnisse ab 100 Kopien aufwärts (das entspricht 166 Eingangspattern für die SFA) nur noch geringfügig besser. Die verrauschten Kopien (*noise*) liefern am schnellsten die besten Ergebnisse. Die gekürzten Kopien (*shorted*) liefern bei wenigen Kopien die schlechtesten Ergebnisse, werden aber ab $\sim$ 300 Kopien aufwärts vergleichbar mit den verrauschten Kopien. Ein ähnliches Ergebnis bringt auch die Kombination von gekürzten und rotierten Kopien (*shorted+rotation*). Diese konvergieren aber noch schneller in den unteren Bereich als die Rotation(*rotation*) oder das Kürzen einzeln betrachtet.

In einer weiteren Untersuchung wurde verglichen, wie sich die einzelnen PB-Verfahren in dem Fall verhalten, wenn die Daten, die von zuvor nicht im Training vorhandenen waren zum Testen verwendet werden.
Hierzu wurden zum Training die Daten der Person ID 122 mittels PB erweitert. Anschließend wurde mit den Originaldaten der Personen ID 121 und 123 getestet. Wie der grafische Vergleich

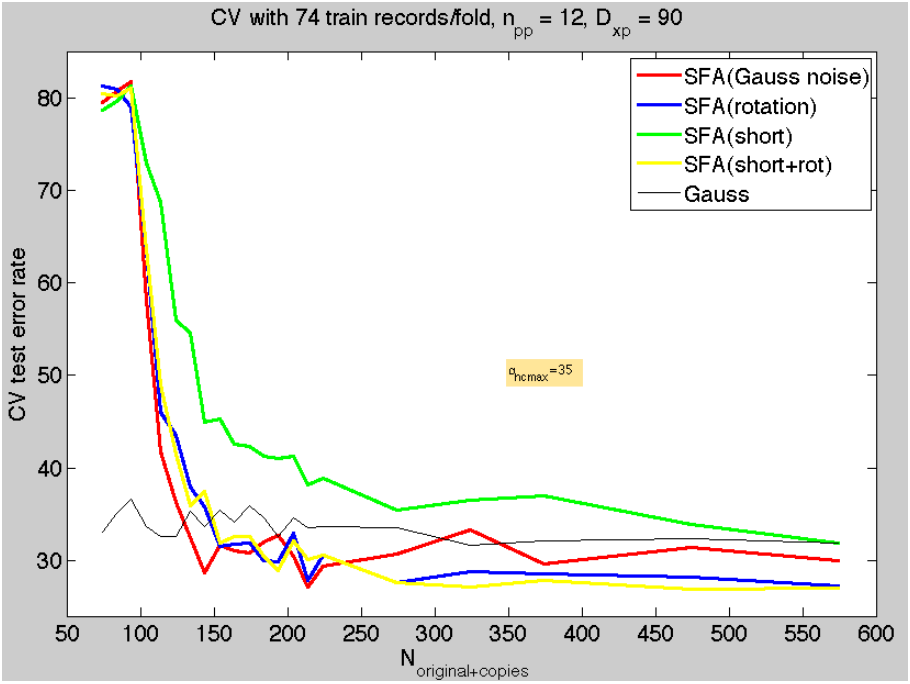

Abb. 2.10.: *Vergleich Bootstrap-Verfahren Rauschen, Drehen, Kürzen, Kürzen+Drehen über 0-500 Kopien + 74 Originaldaten der Person ID 122 als Trainingsdaten und den Originaldaten der Personen ID 121 und 123 zum Testen angewandt mit SFA und exemplarisch einmal mit Gauss*

in Abb. 2.10 zeigt, verbessern sich auch hier die Ergebnisse mit steigender Anzahl an Kopien. Die Fehlerrate ist insgesamt wesentlich höher als in der vorherigen Untersuchung mit den Daten nur einer Person für Training und Test, allerdings zeigen hier die Rotation und die Kombination von Rotation+Kürzen der Daten bessere Resultat als PB durch Verrauschen oder Kürzen allein. Diese Tatsache liegt möglicherweise darin begründet, dass durch die Rotation stärker Variationen der Gesten erzeugt werden, die eventuell mehr den Gesten einer anderen Person auch entsprechen könnten. Bei der Gestendurchführung wird die Wiimote von Person zu Person unterschiedlich gehalten. Eine Rotation der Daten entspricht deshalb eher einer natürlichen Variation, die durch die unterschiedliche Haltung der Wiimote begründet ist.

Werden zum Training die Daten einer Person mit PB verwendet und die Daten aller anderen Personen zum Testen verwendet, werden die Ergebnisse allerdings schlechter als zuvor mit nur zwei Personen. Die Ergebnisse dieser Untersuchung über alle Personen in 10 Läufen gemittel sind in Abb. 2.11 dargestellt. Insgesamt zeigt das Verfahren über Kürzen und Rotieren die besten Ergebnisse wenn die Anzahl der Kopien hoch ist. Mit weniger Kopien zeigt das Verfahren über Verrauschen im Verhältnis die besten Ergebnisse. Es nähert sich aber bei höherer Anzahl an Kopien dem Rotations-Verfahren an.

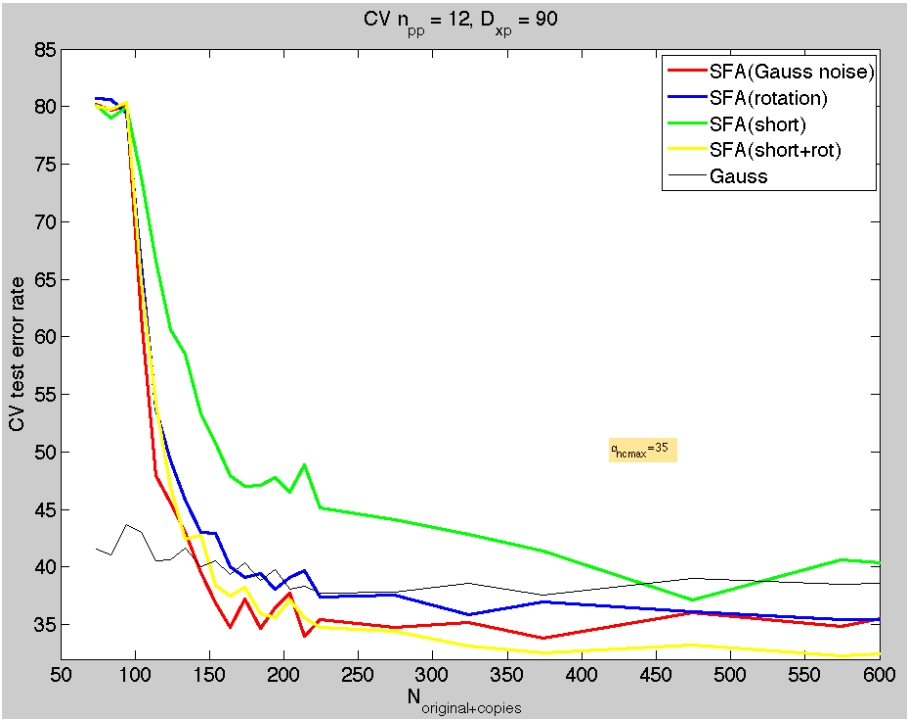

Abb. 2.11.: *Vergleich Bootstrap-Verfahren Rauschen, Drehen, Kürzen, Kürzen+Drehen über 0-600 Kopien + Originaldaten der jeweiligen Person als Trainingsdaten und den Originaldaten aller anderen Personen bei der Klassifizierung mit SFA und exemplarisch einmal mit Gauss.*

2.4.4. Test III - Erkennung der Gesten personenabhängig

Bei anfänglichen Klassifizierungstests mit der SFA zur personenabhängigen Gestenerkennung hat sich herausgestellt, dass die SFA ein sogenanntes "rank deficiency"-Problem (siehe Abschnitt 2.4.2) hat, wenn zu wenig Trainingsdaten vorhanden sind, was zu massiv hohen Fehlerraten von $\sim 80\%$ führte. Eine Möglichkeit, dieses Problem zu umgehen, ist, die Eingangsdaten durch PCA stärker zu reduzieren. Wie aber aus der Tabelle 2.3 zu erkennen ist, ist die Erkennung bei zu geringen Eingangsdimensionen auch nicht mehr optimal.
Um dieses Problem zu beheben, ohne die Eingangsdimensionen zu stark einzuschränken und ohne nochmals neue Daten erheben zu müssen, wird das zuvor beschriebene PB angewendet. Wie die Untersuchungen zu PB ergeben haben, werden die besten Ergebnisse mit 250 verrauschten Kopien erreicht. Aus diesem Grund werden bei der personenabhängigen Klassifizierung mit SFA jeweils 250 verrauschte Kopie aus den Originaltrainingsdaten erzeugt und das Trainingsset mit diesen ergänzt. Das Testset besteht ausschließlich aus Originaldaten.
Getestet wurde mit 10-maliger Durchführung der 10fachen CV pro Person, d.h. als Trainings- und als Testset wurden die Gesten der selben Person verwendet. Die Ergebnisse wurden pro Person gemittelt und dieser Mittelwert $\pm$ der Standardabweichung ist für SFA+PB, Gauss+PB und HMM in der Tabelle 2.5 dargestellt. *numtrn* ist die Größe des jeweiligen Trainingssets.

Tab. 2.5.: *Ergebnisse der personenabhängigen Klassifizierung mit SFA, Gauss, HMM*

Person	SFA + PB	Gauss + PB	*numtrn* (PB)	HMM	*numtrn*
121	(4.493 ± 0.867)%	(11.159 ± 1.912)%	312	(32.17 ± 3.03)%	62
122	(1.622 ± 1.471)%	(3.919 ± 1.417)%	316	(44.19 ± 2.77)%	66
123	(1.273 ± 1.294)%	(0.545 ± 0.926)%	299	(13.64 ± 3.75)%	50
124	(5.357 ± 2.174)%	(4.107 ± 1.786)%	300	(18.93 ± 1.43)%	50
125	(1.231 ± 0.684)%	(1.846 ± 0.684)%	308	(26.31 ± 2.00)%	59
126	(0.755 ± 1.391)%	(0.943 ± 1.406)%	297	(22.45 ± 2.86)%	48
127	(2.985 ± 1.049)%	(4.925 ± 0.760)%	310	(35.67 ± 1.82)%	60
128	(0.857 ± 0.778)%	(2.000 ± 0.778)%	313	(20.71 ± 1.60)%	63
129	(4.821 ± 1.994)%	(11.607 ± 2.841)%	300	(21.43 ± 1.96)%	50
130	(1.457 ± 0.721)%	(5.033 ± 0.819)%	385	(32.58 ± 2.07)%	135
mean	(**2.4851** ± 1.7787)%	(**4.6084** ± 3.9055)%		(**26.81** ± 9.21)%	

Die Ergebnisse der SFA variieren zwischen 0.76% und 5.36%. Über alle Personen gemittelt ist das Ergebnis mit 2.49% besser als das der personenübergreifenden Klassifizierung mit SFA (Test IV) mit 2.71% bei gleicher Reduktionsdimension (*pprange* = 12). Daten einzelner Personen, z.B. die der Personen 126 und 128, werden dabei recht gut, andere im Verhältnis weniger gut klassifiziert.
Die Ergebnisse der personenabhängigen Klassifizierung mit HMM variiert stark von Person zu Person zwischen 13% und 44%. Die Standardabweichungen innerhalb der CV waren bereits jeweils sehr hoch (die Ergebnisse der einzelnen Läufe sind im Anhang A.2 zu finden).

Um eine Aussage darüber treffen zu können, welche Gesten mit welchen am häufigsten verwechselt werden und welche Gesten generell am häufigsten missklassifziert werden, ist exemplarisch pro Person (ID 121-130) eine Konfusionsmatrix dargestellt, für SFA in Tabelle 2.6 und für HMM in Tabelle 2.7.

Tab. 2.6.: *Konfusionsmatrizen der Personen ID 121-130 bei personenabhängiger Klassifizierung mit der SFA (CV)*

Person 121	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	23	0	0	0	1
throw	0	10	0	0	0
frisbee	0	1	10	1	0
bowling	0	0	2	11	0
z	0	0	0	0	10
ClasseErrors	0.0%	9.1%	16.7%	8.3%	9.1%

Person 122	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	19	0	0	0	0
throw	0	20	0	0	0
frisbee	0	0	10	0	0
bowling	1	0	0	14	0
z	0	0	0	0	10
ClasseErrors	5.0%	0.0%	0.0%	0.0%	0.0%

Person 123	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	11	0	0	0	0
throw	0	11	0	0	0
frisbee	0	0	10	0	0
bowling	0	0	1	11	0
z	0	0	0	0	11
ClasseErrors	0.0%	0.0%	9.1%	0.0%	0.0%

Person 124	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	11	0	0	0	0
throw	1	11	0	0	0
frisbee	0	0	10	1	0
bowling	0	0	1	10	0
z	0	0	0	0	11
ClasseErrors	8.3%	0.0%	9.1%	9.1%	0.0%

Person 125	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	19	0	0	0	0
throw	0	11	0	0	0
frisbee	0	0	11	0	0
bowling	0	0	1	9	0
z	0	0	0	1	13
ClasseErrors	0.0%	0.0%	8.3%	10.0%	0.0%

Person 126	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	11	0	0	0	1
throw	0	11	0	0	0
frisbee	0	0	11	0	0
bowling	0	0	0	10	0
z	0	0	0	0	9
ClasseErrors	0.0%	0.0%	0.0%	0.0%	10.0%

Person 127	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	10	0	0	0	0
throw	0	17	0	0	0
frisbee	0	1	9	1	0
bowling	0	0	0	17	0
z	0	0	1	0	11
ClasseErrors	0.0%	5.6%	10.0%	5.6%	0.0%

Person 128	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	19	0	0	0	0
throw	0	12	0	0	0
frisbee	0	0	19	1	0
bowling	0	0	0	9	0
z	0	0	0	0	10
ClasseErrors	0.0%	0.0%	0.0%	10.0%	0.0%

Person 129	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	9	0	0	0	0
throw	0	15	0	0	0
frisbee	0	1	10	0	0
bowling	0	0	0	10	0
z	1	0	0	0	10
ClasseErrors	10.0%	6.2%	0.0%	0.0%	0.0%

Person 130	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	31	0	0	0	0
throw	0	28	0	1	0
frisbee	0	0	31	1	0
bowling	0	0	0	29	0
z	0	0	0	0	30
ClasseErrors	0.0%	0.0%	0.0%	6.5%	0.0%

Innerhalb der personenabhängigen Klassifizierung mit SFA werden insgesamt die *frisbee*- und *bowling*-Geste am schlechtesten klassifiziert. *circle* und *throw* liegen im mittleren Bereich. Das beste Ergebnis bei der Wiedererkennung wird bei der *z*-Geste erzielt.
Eine Betrachtung der HMM-Konfusionsmatrizen in Tabelle 2.7 zeigt, dass hier keine allgemeingültige Aussage zur Missklassifikation über alle Personen möglich ist. Der Fehler pro Gestenklasse variiert von Person zu Person.
Auch insgesamt variieren die Ergebnisse jedes betrachteten Klassifizierers sehr stark von Person zu Person. Die Ergebnisse der für jede Person mit den Klassifizierern SFA, Gauss und HMM sind in der Abb. 2.12 nochmals gegenübergestellt. SFA liefert mit einer durchschnittlichen Fehlerrate von 2.5% das beste Ergebnis, gefolgt von dem Gaussklassifizierer mit 4.6%. Am schlechtesten schneidet HMM mit 26.8% ab.

Tab. 2.7.: *Konfusionsmatrizen der Personen ID 121-130 bei personenabhängiger Klassifizierung mit HMM (CV)*

Person 121	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	5	0	0	0	0
throw	0	9	0	0	0
frisbee	3	1	11	2	0
bowling	0	1	0	9	0
z	15	0	0	0	11
0% probability	0	0	1	1	0
ClasseErrors	78,26%	18,18%	8,33%	25,00%	0,00%

Person 122	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	19	8	1	11	8
throw	0	11	0	0	0
frisbee	1	1	6	0	0
bowling	0	0	0	3	0
z	0	0	3	0	2
0% probability	0	0	0	0	0
ClasseErrors	5,00%	45,00%	40,00%	78,57%	80,00%

Person 123	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	5	0	0	0	0
throw	0	11	0	0	0
frisbee	4	0	11	1	0
bowling	2	0	0	9	0
z	0	0	0	0	10
0% probability	0	0	0	1	1
ClasseErrors	54,55%	0,00%	0,00%	18,18%	9,09%

Person 124	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	10	0	2	0	2
throw	0	10	0	2	0
frisbee	2	0	9	2	0
bowling	0	1	0	7	0
z	0	0	0	0	9
0% probability	0	0	0	0	0
ClasseErrors	16,67%	9,09%	18,18%	36,36%	18,18%

Person 125	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	17	0	0	0	0
throw	0	4	1	0	1
frisbee	2	0	11	3	0
bowling	0	7	0	4	1
z	0	0	0	3	11
0% probability	0	0	0	0	0
ClasseErrors	10,53%	63,64%	8,33%	60,00%	15,38%

Person 126	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	2	0	0	0	0
throw	0	10	0	0	0
frisbee	9	0	10	0	0
bowling	0	0	0	9	0
z	0	0	0	0	9
0% probability	0	1	1	1	1
ClasseErrors	81,82%	9,09%	9,09%	10,00%	10,00%

Person 127	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	10	0	0	0	1
throw	0	17	2	1	1
frisbee	0	0	8	15	3
bowling	0	0	0	2	0
z	0	0	0	0	6
0% probability	0	1	0	0	0
ClasseErrors	0,00%	5,56%	20,00%	88,89%	45,45%

Person 128	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	14	0	0	0	0
throw	0	12	1	0	0
frisbee	5	0	18	10	0
bowling	0	0	0	0	0
z	0	0	0	0	10
0% probability	0	0	0	0	0
ClasseErrors	26,32%	0,00%	5,26%	100,00%	0,00%

Person 129	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	9	0	0	0	3
throw	0	15	1	0	0
frisbee	1	0	8	2	1
bowling	0	0	1	8	0
z	0	0	0	0	6
0% probability	0	1	0	0	0
ClasseErrors	10,00%	6,25%	20,00%	20,00%	40,00%

Person 130	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	8	0	0	1	0
throw	1	28	0	0	0
frisbee	22	0	28	8	14
bowling	0	0	1	22	0
z	0	0	2	0	16
0% probability	0	0	0	0	0
ClasseErrors	74,19%	0,00%	9,68%	29,03%	46,67%

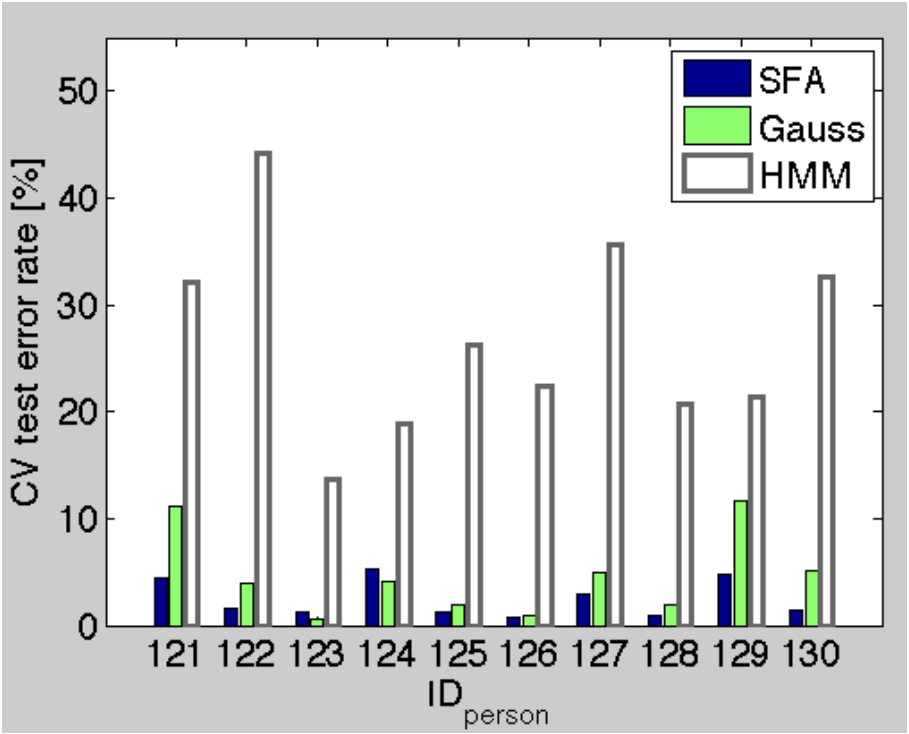

Abb. 2.12.: *Grafischer Vergleich der personenabhängigen Klassifizierungen mit SFA, Gauss und HMM über alle Einzelpersonen*

2.4.5. Test IV - Erkennung der Gesten personenübergreifend

Zum Testen der personenübergreifenden Klassifizierung wurden zufällig Gesten aus der gesamten Datenmenge, bestehend aus den Gesten aller 10 Personen, ausgewählt. Getestet wurden jeweils in 10 Läufen mit 10facher CV über die insgesamt 716 Gestendaten aller Personen. Dazu wurden jeweils ca. 644 Daten als Trainingsset gewählt und die restlichen Daten zum Testen verwendet. Die 10 Teilmengen wurden dabei zufällig aus den gesamten Daten gewählt.

Die Ergebnisse der 10 Testläufe wurden gemittelt. Mittelwert ± Standardabweichung der Klassifizierungstests mit SFA, Gauss und HMM über die Daten aller Personen sind in Tabelle 2.8 dargestellt. Für diesen Vergleich wurden jeweils zufällige Trainings- und Testsets ausgewählt. *pprange* 12 bzw. 15 besagt, dass die PCA den Eingangsvektor auf 12 bzw. 15 Eingangsdimensionen reduziert (oN = ohne Amplitudennormierung, N = mit Amplitudennormierung (siehe Vorverarbeitungsschritt der Amplitudennormierung in Abschnitt 2.3.3)).

Tab. 2.8.: *Ergebnisse der personenübergreifenden Klassifizierung mit SFA, Gauss, HMM im Vergleich*

SFA			Gauss	
pprange12 oN	pprange12 N	pprange15 N	oN	N
6.75%±0.46	**2.71%**±0.35	**2.12%**±0.25	**15.44%**±0.47	**14.19%**±0.85

HMM	
original	N
45,39%±5,80	**52,79%**±6,97

Das Ergebnis der SFA-Klassifizierung mit Normierung der Amplitude über die Standardabweichung ist besser als ohne diese Amplitudennormierung. Die Fehlerraten des Gaussklassifizierers liegen mit durchschnittlich 14-15% wesentlich höher als die der SFA, mit und sogar auch ohne Amplitudennormierung. Die Fehlerrate des HMM-Tests ist mit 45,39% im Verhältnis sehr hoch. Getestet wurde HMM ebenfalls mit komplett vorverarbeiteten Daten (Vorverarbeitung siehe 2.3.3), was mit einer durchschnittlichen Fehlerrate von 52,79% (Standardabweichung: 6,97%) ein noch schlechteres Ergebnis lieferte.

Zur Übersicht darüber, dass bestimmte Gesten besser erkannt werden als andere, an welchen Stellen es am häufigsten zu Missklassifikation kommt und wie groß der Fehler pro Klasse ist, ist in Tabelle 2.9 die Konfusionsmatrix des ersten SFA CV-Lauf (pprange12 N) mit kompletter Vorverarbeitung (inklusive Amplitudennormierung) dargestellt. Man sieht, dass die *throw*- und die z-Geste ein bisschen besser erkannt werden als die anderen drei Gesten.

Tab. 2.9.: *Konfusionsmatrix (exemplarisch von einem Testlauf) bei personenübergreifender Klassifizierung mit der SFA unter Verwendung von normierten Daten (10fach CV über alle Daten)*

all Persons	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	160	0	0	0	1
throw	1	147	0	2	0
frisbee	3	2	132	3	1
bowling	0	0	5	132	0
z	2	0	0	0	125
ClasseErrors	3.6%	1.3%	3.6%	3.6%	1.6%

An der Konfusionsmatrix 2.10 über einen HMM-Testlauf mit den Originaldaten erkennt man, dass auch hier die Gesten *throw* und *z* über alle Daten hinweg am besten erkannt wurden. Es gab keine Geste die mit 0% Wahrscheinlichkeit erkannt wurde, d.h. alle Gesten sind mit einer höheren Wahrscheinlichkeit entweder der richtigen oder der falschen Klasse zugeordnet worden. Am schlechtesten und nahezu gar nicht wiedererkannt wurde mit HMM die *bowling*-Geste.

Tab. 2.10.: *Konfusionsmatrix (exemplarisch von einem Testlauf) bei personenübergreifender Klassifizierung mit HMM unter Verwendung der Originaldaten (10fach CV über alle Daten)*

all Persons	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	52	0	19	1	14
throw	2	133	9	19	0
frisbee	35	8	98	112	8
bowling	0	5	4	4	1
z	77	3	7	1	104
0% probability	0	0	0	0	0
ClasseErrors	68,67%	10,74%	28,47%	97,08%	18,11%

2.4.6. Test V - Erkennung der Gesten einer neuen Person

Für die Erkennung der Gesten einer neuen Person, deren Daten zuvor nicht zum Training des Modells verwendet wurde, wird mit der leave-one-out Cross-Validation getestet. Zum Training des Modells werden die Daten von 9 Personen verwendet. Das Testset besteht dann aus den Daten der fehlenden 10. Person. Dies wird für jede Person einmal durchgeführt.

Abb. 2.13 zeigt die Ergebnisse dieser CV für SFA, Gaussklassifizierer, Random Forest [9] und HMM. Die Fehlerraten sind erwartungsgemäß insgesamt höher als bei der personenübergreifenden Klassifizierung, wo in den zufällig gewählten Trainingssets mit hoher Wahrscheinlichkeit bereits Pattern der Person, deren Daten später zum Testen verwendet werden, enthalten sind.

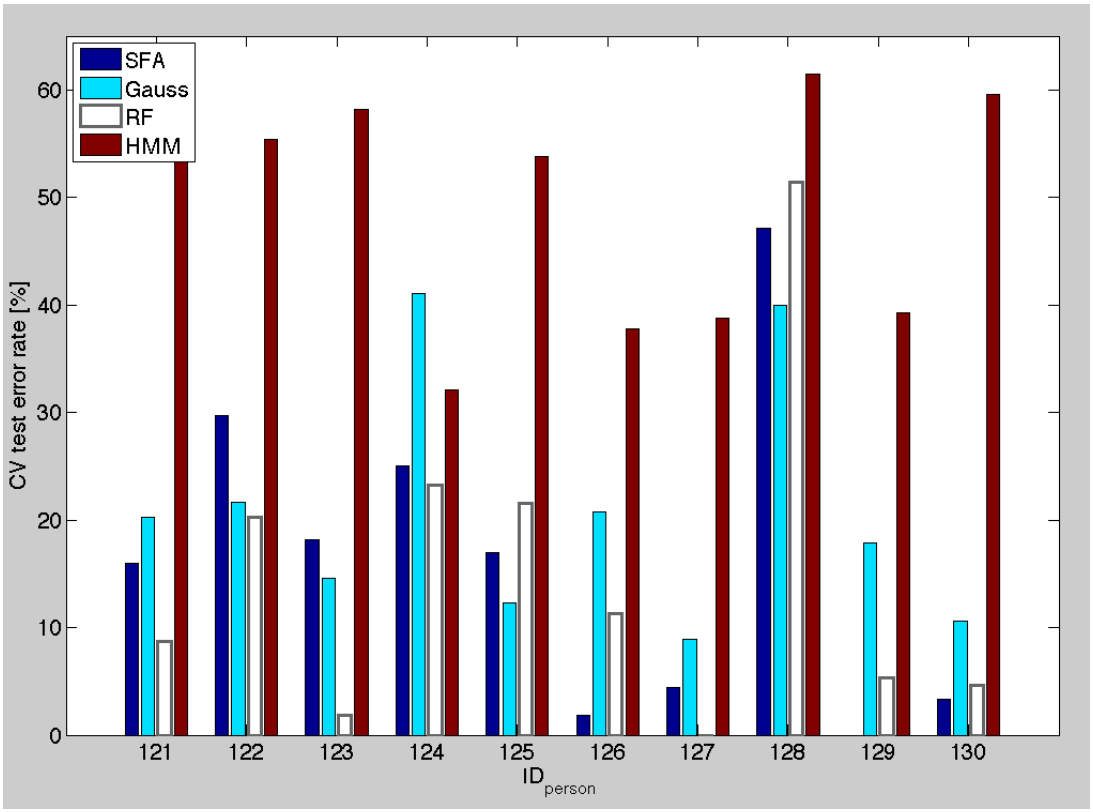

Abb. 2.13.: *Grafischer Vergleich über alle Einzelpersonen - Erkennung der Gesten einer neuen Person mit SFA, Gauss, RF, HMM*

In der Tabelle 2.11 sind die Ergebnisse der CV dargestellt. Die PersonID gibt dabei jeweils die Person an, deren Daten als Testset verwendet wurden. Der Durchschnitt über alle Läufe zeigt, dass RF gefolgt von SFA die besten Ergebnisse liefert. HMM liefert mit Fehlerraten von 32%-61% die schlechtesten Klassifizierungsergebnisse. Auffällig ist die große Variation zwischen den Personen. Mit allen Klassifizierern variieren die Fehlerraten sehr stark von Person zu Person. Die Gesten einiger Personen, z.B. ID 127, 129, 130, lassen sich, obwohl sie nicht trainiert wurden, gut klassifizieren, während andere, wie z.B. ID 128, für alle Klassifizierer schwierig klassifizierbar sind. Vermutlich haben die Gesten dieser Person andere Charakteristika oder dies ist auf die starken Variationen, die bereits innerhalb der Gesten dieser Person auftreten, zurückzuführen. Das müsste aber genauer untersucht werden.

Tab. 2.11.: *Ergebnisse aller Einzelpersonen für die Klassifizierung der Gesten einer neuen Person mit SFA, Gauss, RF, HMM*

PersonID	SFA	Gauss	RF	HMM
121	15.942%	20.290%	8.696%	53.620%
122	29.730%	21.622%	20.270%	55.410%
123	18.182%	14.545%	1.818%	58.180%
124	25.000%	41.071%	23.214%	32.140%
125	16.923%	12.308%	21.538%	53.850%
126	1.887%	20.755%	11.321%	37.740%
127	4.478%	8.955%	0.000%	38.810%
128	47.143%	40.000%	51.429%	61.430%
129	0.000%	17.857%	5.357%	39.290%
130	3.311%	10.596%	4.636%	59.600%
mean(all)	(**16.26**±14.88)%	(**20.80**±11.28)%	(**14.83**±15.34)%	(**50.84**±10.39)%

[9]Die Ergebnisse von RF sind aus Koch et al. [KKH10] übernommen.

Die SFA-Konfusionsmatrix in Tabelle 2.12 mit der Summe der CV-Einzelergebnisse zeigt, dass insgesamt die *z*-Gesten bei Tests mit Gesten einer neuen Person mit 10% am besten klassifiziert wurden. Vermutlich weil diese Geste nicht so viele andere Ausprägungen zulässt wie z.B. die *frisbee*- oder *bowling*-Geste, die mit über 20% Fehlerrate bei Klassifizierung der Gesten einer neuen Person am schlechtesten abschneidet. Sie läßt aber auch wesentlich mehr Variationen zu als eine genauer definierte *z*-Geste.

Tab. 2.12.: *Konfusionsmatrix Klassifizierung einer neue Person mit SFA, Summe CV*

all Persons	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	142	3	7	0	12
throw	0	125	4	2	0
frisbee	10	16	104	26	1
bowling	0	5	22	109	0
z	14	0	0	0	114
ClassErrors:	14.5%	16.1%	24.1%	20.4%	10.2%

Die HMM-Konfusionsmatrix 2.13 über alle Einzeltests zusammen zeigt, dass insgesamt bei diesem Test die *throw*- und die *z*-Geste mit 18-19% am besten klassifiziert wurden. Alle anderen zeigen schlechte bis sehr schlechte Ergebnisse bei der Klassifizierung mit HMM.

Tab. 2.13.: *Konfusionsmatrix Klassifizierung einer neue Person mit HMM, Summe CV*

all Persons	true class				
found ⇓	circle	throw	frisbee	bowling	z
circle	49	4	10	4	19
throw	6	120	12	27	0
frisbee	36	10	79	102	4
bowling	3	3	0	1	1
z	72	12	36	3	103
ClassErrors:	70.48%	19.46%	42.34%	99.27%	18.90%

Insgesamt lässt sich für beide Verfahren, SFA und HMM, feststellen, dass manche Gesten von unterschiedlichen Personen relativ ähnlich ausgeführt werden, z.B. die *z*-Geste, und dementsprechend auch besser erkannt werden. Weniger spezielle Gesten lassen größere Variationen zu. Dadurch sind Gesten einer neuen Person schwieriger klassifizierbar.

Eine Untersuchung des umgekehrten Falls, wenn nur die Daten einer Person zum Training verwendet und mehrere ungesehene Personen auf diesen Daten getestet wurden, ergab für die Klassifizierung mit der SFA und PB mit 250 verrauschten Kopien eine durchschnittliche Fehlerrate von $\sim 39\%$. Die Einzelergebnisse pro Person sind der Tabelle 2.14 zu entnehmen. Die Ergebnisse sind insgesamt erwartungsgemäß wesentlich schlechter, als wenn beim Training die Gesten von mehreren Personen verwendet werden. Diese Testläufe bestand daraus, die Daten jeder einzelnen Person zuerst zum Trainieren der SFA inklusive PB *noise* 250 zu verwenden. Anschließend jeweils die Origrinaldaten aller anderen Personen zu testen verwendet. Für jede Person wurden 10 Testläufe durchgeführt und die Ergebnisse gemittelt.

Tab. 2.14.: *Ergebnisse aller Einzelpersonen für die Klassifizierung der Gesten neuer Personen mit SFA + PB (noise 250), wenn zum Training nur die Daten einer Person verwendet werden.*

PersonID	numtrn	SFA + PB
121	319	43.045%
122	324	35.389%
123	305	33.979%
124	306	33.758%
125	315	44.946%
126	303	34.646%
127	317	39.938%
128	320	65.279%
129	306	34.394%
130	401	28.973%
all		39.212%

Ohne PB kam es bei der Klassifizierung mit der SFA von ungesehenen Personen zu sehr hohen Fehlerraten. Deswegen wurde PB eingesetzt. Alle verwendeten PB-Verfahren lieferten insgesamt mit 250 Kopien bessere Ergebnisse die in Abb. 2.14 dargestellt sind.

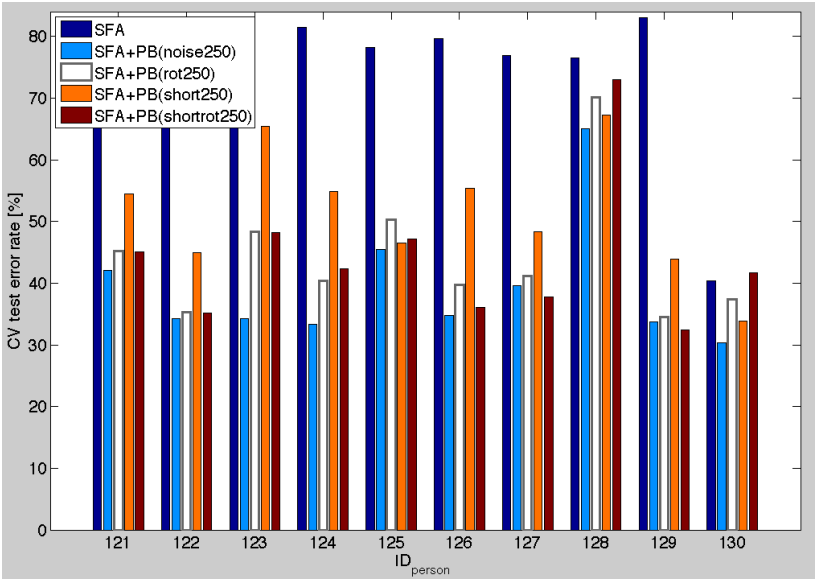

Abb. 2.14.: *Vergleich Klassifizierung neuer Personen mit SFA ohne PB und mit PB (noise250, rotation250, short250, short+rotation250)*

2.4.7. Test VI - Minimales Trainingsset

Da in realen Umgebungen häufig nur wenige Trainingsdaten vorliegen, soll in dieser Untersuchung betrachtet werden wie sich die Klassifizierung mit der SFA verhält, wenn nur wenige Trainingsdaten vorhanden sind. Bei diesem Test wurden als Trainingsdaten $n_{original}$ Originalpattern (a. einer Person, b.aller Personen) pro Gestenklasse verwendet und diese um eine festen Anzahl an künstlichen verrauschten Pattern ergänzt ($ncopies = 250$). Das Testset bestand jeweils aus den restlichen Originalpattern (a.einer Person, b.aller Personen).

Um zu testen, wie viele Originaldaten notwendig sind, wurden, zum Einen über jede Person einzeln eine unterschiedliche Anzahl von Eingangsdaten bei einer Klassifizierung mit der SFA (und Gauss) getestet, deren gemittelte Ergebnisse *mean* n_{pp} in Abb. 2.15 grafisch dargestellt sind. Zum anderen wurde dieser Test nochmals über alle Daten durchgeführt n_{all}. $n_{original}$ entspricht dabei der Anzahl der Trainingsdaten pro Gestenklasse. Die Originaltrainingsdaten wurden dabei zufällig aus den vorhandenen Daten gewählt und jeweils mit 250 verrauscht Kopie ergänzt, die aus den Originaldaten erzeugt wurden. Die Detailergebnisse in Tabelle 2.15 entsprechen dem Mittelwert aus 10 mal 10 Testläufen.

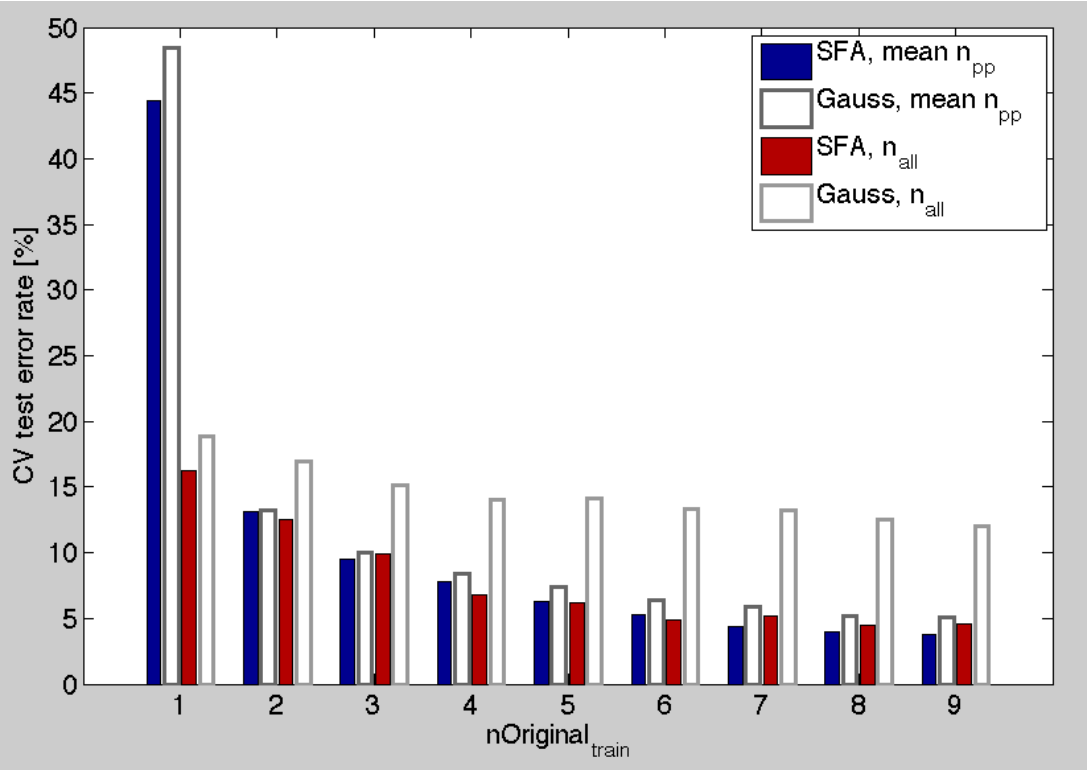

Abb. 2.15.: *Ergebnisse Minimales Trainingsset - Random Test mit n Originaldaten pro Geste (a.pro Person n_{pp} gemittelt, b.für alle Personen n_{all}), Anzahl der verwendeten Pattern ist (n * 5 Gesten) + 250 verrauschte Kopien*

Die Ergebnisse zeigen, dass sich mit steigender Anzahl an Originaldaten die Klassifizierungsergebnisse für alle Verfahren verbessern, allerdings nicht an die Ergebnisse von Mäntyjärvi et al. [MKKK04] heranreichen. Ein direkter Vergleich mit dem Ansatz von Mäntyjärvi et al., die mit 2 Originalpattern pro Gestenklasse gute Ergebnisse erzielen, ist in diesem Fall jedoch nicht möglich. 2 Originaldaten reichen für eine Klassifizierung mit der SFA aber auf jeden Fall nicht generell aus, um an diese Ergebnisse heranzukommen. Für Person 123 wird mit 2 Originaldatensätzen

pro Gestenklasse (das entspricht in diesem Fall insgesamt 10 Originaltrainingspattern) bereits eine mittlere Fehlerrate von 3.64% erreicht. Andere Personen erreichen aber selbst mit der maximal getesteten Anzahl von 9 Originaldaten pro Geste (insgesamt 45 Originaltrainingspattern) nicht annähernd dieses Ergebnis.

Tab. 2.15.: *Ergebnisse Minimales Trainingsset - Randomtest mit variabler Anzahl von Originaldaten $n_{original}$ pro Geste + 250 Kopien als Trainingsset*

	$n_{original}$ = 1 (5+250)		$n_{original}$ = 2 (10+250)		$n_{original}$ = 3 (15+250)	
PersonID	**SFA**	Gauss	**SFA**	Gauss	**SFA**	Gauss
121	(66.484±3.375)%	(67.234±3.830)%	(20.051±2.560)%	(20.661±2.076)%	(14.741±1.695)%	(15.963±1.930)%
122	(58.348±2.099)%	(56.739±3.640)%	(21.391±3.458)%	(22.062±3.089)%	(14.847±3.169)%	(15.102±2.709)%
123	(35.200±3.205)%	(43.600±2.592)%	(3.644±0.919)%	(2.978±1.240)%	(2.225±0.856)%	(1.400±0.660)%
124	(58.098±2.746)%	(58.196±2.139)%	(19.109±2.901)%	(19.565±2.856)%	(12.537±2.165)%	(14.244±1.794)%
125	(36.833±3.256)%	(42.100±5.063)%	(5.909±1.412)%	(6.164±1.173)%	(4.240±0.952)%	(4.200±0.994)%
126	(35.917±4.633)%	(42.646±5.306)%	(8.000±2.366)%	(5.651±1.546)%	(5.000±1.638)%	(4.158±1.590)%
127	(29.242±1.093)%	(31.210±3.899)%	(11.912±2.173)%	(12.754±2.623)%	(9.942±1.813)%	(10.365±1.441)%
128	(39.615±3.546)%	(45.815±3.882)%	(8.683±2.532)%	(9.183±2.538)%	(5.109±2.120)%	(5.636±1.757)%
129	(35.667±2.886)%	(42.451±2.751)%	(20.087±1.688)%	(21.761±1.526)%	(16.732±1.778)%	(19.463±1.935)%
130	(48.884±2.719)%	(54.438±4.162)%	(12.191±1.865)%	(11.652±1.439)%	(9.485±1.144)%	(9.324±1.089)%
mean(person)	**44.429%**	48.442%	**13.098%**	13.243%	**9.486%**	9.986%
	$n_{original}$ = 1 (50+250)		$n_{original}$ = 2 (100+250)		$n_{original}$ = 3 (150+250)	
all	(16.231±2.538)%	(18.844±2.336)%	(12.484±2.516)%	(16.948±1.563)%	(9.912±1.674)%	(15.159±2.623)%

	n_{orig} = 4 (20+250)		n_{orig} = 5 (25+250)		n_{orig} = 6 (30+250)	
PersonID	**SFA**	Gauss	**SFA**	Gauss	**SFA**	Gauss
121	(12.776±1.626)%	(14.327±1.635)%	(10.341±1.602)%	(12.886±1.124)%	(9.231±2.551)%	(12.513±2.228)%
122	(12.056±3.174)%	(11.444±3.253)%	(9.041±2.274)%	(9.653±2.151)%	(7.318±1.716)%	(6.886±1.528)%
123	(1.943±0.693)%	(1.314±0.638)%	(1.767±0.575)%	(0.933±0.491)%	(1.000±0.571)%	(0.400±0.344)%
124	(10.278±2.351)%	(10.722±2.605)%	(7.484±1.349)%	(8.290±2.096)%	(5.885±1.030)%	(6.769±1.855)%
125	(3.844±0.924)%	(3.844±0.884)%	(2.875±0.881)%	(3.050±0.885)%	(2.857±1.062)%	(2.143±0.727)%
126	(3.909±1.539)%	(3.030±1.295)%	(3.143±1.277)%	(3.071±0.692)%	(3.217±1.127)%	(2.478±0.967)%
127	(7.660±0.975)%	(8.745±1.096)%	(6.119±1.556)%	(7.738±1.610)%	(5.649±1.653)%	(6.919±1.466)%
128	(3.660±1.176)%	(4.720±1.330)%	(2.622±1.611)%	(3.333±1.686)%	(1.625±0.907)%	(2.725±1.472)%
129	(14.222±2.370)%	(18.083±1.737)%	(11.419±2.405)%	(16.548±2.494)%	(9.423±3.284)%	(15.769±2.967)%
130	(7.992±0.630)%	(8.191±0.948)%	(7.897±1.006)%	(8.087±0.948)%	(6.397±0.897)%	(7.372±0.977)%
mean(person)	**7.834%**	8.442%	**6.271%**	7.359%	**5.260%**	6.397%
	$n_{original}$ = 4 (200+250)		$n_{original}$ = 5 (250+250)		$n_{original}$ = 6 (300+250)	
all	(6.802±1.443)%	(14.050±1.380)%	(6.223±0.905)%	(14.120±2.565)%	(4.904±1.043)%	(13.293±2.034)%

	n_{orig} = 7 (35+250)		n_{orig} = 8 (40+250)		n_{orig} = 9 (45+250)	
PersonID	**SFA**	Gauss	**SFA**	Gauss	**SFA**	Gauss
121	(8.059±1.333)%	(13.147±1.332)%	(6.759±1.126)%	(12.207±1.955)%	(5.897±1.857)%	(11.517±2.790)%
122	(6.231±1.847)%	(6.744±1.661)%	(4.971±1.557)%	(5.471±1.622)%	(5.618±2.190)%	(6.647±2.672)%
123	(1.050±0.631)%	(0.500±0.430)%	(1.333±0.994)%	(0.267±0.491)%	(1.667±1.206)%	(0.267±0.363)%
124	(5.524±2.210)%	(5.190±2.553)%	(5.375±1.430)%	(5.188±2.219)%	(5.062±2.023)%	(4.625±2.017)%
125	(1.700±0.387)%	(1.967±0.730)%	(1.600±0.526)%	(1.400±0.916)%	(1.240±0.643)%	(1.680±0.479)%
126	(1.667±0.916)%	(1.389±1.078)%	(1.308±0.859)%	(1.077±1.095)%	(1.385±0.837)%	(1.308±1.381)%
127	(4.000±1.430)%	(5.844±1.336)%	(4.000±1.299)%	(5.444±1.438)%	(3.593±1.235)%	(4.704±1.235)%
128	(1.171±0.702)%	(2.857±1.398)%	(1.233±1.148)%	(2.900±1.615)%	(0.933±0.773)%	(1.967±0.630)%
129	(9.190±1.908)%	(14.429±1.606)%	(8.125±3.213)%	(12.250±3.598)%	(7.375±1.959)%	(12.250±2.883)%
130	(5.578±0.731)%	(6.569±0.899)%	(4.757±0.605)%	(5.865±1.079)%	(5.252±0.839)%	(6.081±0.572)%
mean(person)	**4.417%**	5.863%	**3.946%**	5.207%	**3.802%**	5.105%
	n_{orig} = 7 (350+250)		n_{orig} = 8 (400+250)		n_{orig} = 9 (450+250)	
all	(5.219±1.073)%	(13.224±1.579)%	(4.494±0.979)%	(12.532±1.502)%	(4.586±1.236)%	(12.030±1.607)%

2.5. Diskussion

- Wie in der Untersuchung zur Auswirkung der Reduktion 2.4.2 gezeigt wurde, kommt es bei der SFA zu Overfitting-Problemen, wenn zu wenig Trainingsdaten vorhanden sind. Aus diesem Grund war es nötig, beim Test zur personenabhängigen Klassifizierung (Test III) die Trainingsdaten durch zusätzliche künstliche Daten zu ergänzen. Die Ergebnisse der SFA für personenabhängige Klassifizierung zeigen dann aber weniger Klassifizierungsfehler auf den Testdaten als die Ergebnisse des Vergleichs mit dem Gaussklassifizierer oder HMM.

- Der Test zur personenübergreifenden Klassifizierung hat gezeigt, dass SFA dabei zu guten Ergebnissen kommt, während HMM an dieser Stelle hohe Fehlerraten aufweist.

- Werden die Daten von Personen unterschiedlichen Geschlechts, Alters und Händigkeit für Training und Tests herangezogen, erhält man Ergebnisse heterogener Art. Die Klassifizierung der Gesten einer neuen Person mit SFA ist aber im Durchschnitt über alle Tests mit 15-17% Fehlerrate immer noch gut, verglichen damit, dass die Klassifizierung mit HMM überhaupt nicht imstande ist, vergleichbare Ergebnisse hervorzubringen. Das durchschnittliche Ergebnis der SFA-Klassifizierung liegt außerdem immer noch nahezu in dem Bereich, in dem HMM personenabhängig Gesten klassifiziert.

- Die ermittelten Ergebnisse beim Klassifizierungstests mit HMM von Daten einzelner Personen, wie in Tabelle A.1 dargestellt, sind schlechter als die in Schlömer et al. [SPHB08] dargestellten. Mögliche Ursache könnte sein, dass dort teilweise minimal mehr Trainingsdaten zur Verfügung standen (immer 15 pro Person und Geste, hier zumindest 10 pro Person und Geste). Die Gesten sind unterschiedlich. *Circle* und z-Geste sind in beiden Untersuchungen vorhanden. Die verwendete *circle*-Geste dieser Arbeit war schwer klassifizierbar, wo hingegen die z-Geste am häufigsten richtig klassifiziert wurde.

- Die HMM-Vergleichsimplementierung verwendet das WiiGee-Framework wie es bei Poppinga und Schlömer [PS07] verfügbar ist. Trotz mehrfacher Prüfung auf Korrektheit der Implementierung lassen sich weder die Abweichungen zu den Ergebnissen von Schlömer et al. [SPHB08] begründen, noch, dass diese HMM-basierte Gestenerkennung insgesamt zu relativ schlechten Ergebnissen kommt.

- Die ermittelten Ergebnisse beim Test von Daten einer Person, deren Gesten nicht im Trainingsset enthalten waren, zeigen, dass eine Klassifizierung mit SFA (ohne PB) schlechtere Ergebnisse aufweist als ein gängiges Verfahren wie Random Forest, das wie in Koch et al. [KKH10] beschrieben auch mit wenigen Trainingsdaten gute Ergebnisse liefert. Für die Betrachtung des umgekehrten Falls wenn nur die Daten einer Person zum Training und die Daten aller anderen Personen zum Testen verwendet wurden, hat gezeigt, dass hier PB verwendet werden muss um überhaupt brauchbare Ergebnisse zu erzielen. Erwartungsgemäß sind dieser aber schlechter als, wenn die Originaldaten von mehreren Personen zum Training verwendet werden. In weiteren Untersuchungen könnte betrachtet werden wieviele unterschiedliche Personen zum Training notwendig sind um bessere Ergebnisse zu erzielen.

- Die Betrachtung der Bootstrapverfahren hat gezeigt, dass für eine personenabhängige Klassifizierung mit der SFA das PB-Verfahren durch Verrauschen der Daten die besten Ergebnisse erzielt, während für eine Klassifizierung von neuen Personen das Rotationsverfahren und insbesondere die Kombination von Kürzen und Rotieren besser abschneiden. Die Fehlerraten liegen hier allerdings in Bereichen von 30-40% und teilweise höher.
  Im Hinblick darauf, dass die Rotation mehr Variationen hineinbringt, die vermutlich ähnlicher den Daten anderer Personen sind (gegründet dadurch das einzelne Personen die

Wiimote bei der Gestendurchführung unterschiedlich halten), sind mit der Rotation oder auch der Kombination aus Kürzen und Rotieren für die Daten von ungesehen Testperson noch bessere Ergebnisse als bei den Verfahren über Kürzen allein oder Verrauschen zu erwarten. Dies könnte wie zuvor bereits genannt auch hier dadurch weiter untersucht werden, dass die Daten mehrerer Personen zum Training verwendet werden.

- Es wurde ein relativ großes Trainingsset verwendet, wie es nach Liu et al. [LZWV09] notwendig ist, um mit HMM personenübergreifend klassifizieren zu können. Die Ergebnisse zeigen aber, dass zumindest die hier verwendete HMM-Implementierung nicht zur personenübergreifenden Klassifizierung in der Lage ist. In der betrachteten Literatur finden sich auch keine weiteren Angaben, wie gut anderweitig eine personenübergreifende Klassifizierung mit HMM möglich ist.

- Liegen nur sehr wenige Testdaten pro Person vor, so ist es mit der SFA und Parametric Bootstrap trotzdem möglich, personenabhängig wie auch personenübergreifend zu klassifizieren. Diese Erweiterung des Trainingssets um künstliche Daten wird des Öfteren bei der Klassifizierung mit HMM verwendet, z.B. von Mantyjarvi et al. [MKKK04]. Dort allerdings nur für personenabhängige Daten. Mit SFA und drei Originalpattern pro Gestenklasse lassen sich bereits Fehlerraten unter 10% erreichen. Die Ergebnisse sind allerdings nicht direkt mit denen von Mantyjarvi et al. vergleichbar.

3. Segmentierung von Gestensignalen

Mit Segmentierung wird die Trennung von Geste und Nicht-Geste des kontinuierlichen Eingangssignals bezeichnet, wodurch "automatisch" Anfang und Ende einer Geste festgelegt werden sollen. Das Gestensegment kann dabei beliebige Gestendaten enthalten, das Nichtgestesegment entspricht i.d.R. der relativen Ruhelage der Wiimote.
In diesem Kapitel soll nun untersucht werden, ob es mit der SFA möglich ist, Gesten-Segmente von Nicht-Gesten-Segmenten zu trennen. Im Vergleich dazu wurden zwei andere Verfahren ebenfalls getestet. Abschließend werden die Ergebnisse der einzelnen Verfahren verglichen und bewertet.

3.1. State of the art

Bevor Gesten klassifiziert werden können, müssen sie aus einem i.d.R. kontinuierlichen Eingangssignal herausgefiltert werden. Viele Ansätze zur Gestenerkennung markieren daher den Beginn und das Ende einer Geste durch ein externes Signal, z.B. durch das Drücken einer Taste während die Geste ausgeführt wird([KKM+06], [MKKK04], [MS08], [RBA08]). Ein paar Ansätze sehen aber auch eine automatische Segmentierung vor, z.B. [LZWV09] und AILive, LiveMove-Pro [Inc06], [HHH98] und [Pre08].

Viele explizite Segmentierungsmethoden, die in der Literatur genannt werden, treffen die Annahme, dass eine Geste eine Ruheposition zu Beginn und am Ende einer Geste hat. Es wird angenommen, dass in einer solchen Ruheposition bestimmte Messaktivitäten unter einen gewissen Schwellwert fallen. Wenn das passiert, ist eine Gestengrenze gefunden. Dieser Gedanke von Ruhepositionen wird in [McN92] beschrieben. McNeill erklärt, dass eine Geste immer in einem Ruhezustand beginnt und endet. Er beschreibt sogar weitere Ruhepositionen innerhalb einer Geste: Vorbereitung, optionaler Stop vorher, Kern, optionaler Stop nachher, Rückführung. Diese kurzen Stops innerhalb einer Geste können ein potenzielles Problem sein, da durch sie Grenzen innerhalb einer Geste erkannt werden könnten. Weiterhin unterscheiden sich Stops und Ruhepositionen in ihrer Dauer; Stops sind generell kürzer als Ruhezustände [HS06].

Hofmann et al. [HHH98] verwenden in ihren Ansatz eine einfache Segmentierung, in dem Sie einen Grenzwert definieren. Sobald der Beschleunigungsbetrag r (siehe Eq. (3.1)) über diesem Grenzwert ist, wird eine Geste angenommen. Kleinere Phasen unterhalb des Grenzwertes werden dabei trotzdem zur Geste gezählt, wenn sie innerhalb einer angenommenen Geste liegen. Dieser Ansatz ist ähnlich dem Ansatz von Schlömer et al. [SPHB08], die allerdings ihren so genannten Ruhelagefilter (Idle State Filter) nicht zur Segmentierung einsetzen, sondern ihn dazu verwenden, die Eingangsdaten für die Gestenerkennung mit HMM vorzufiltern und dadurch zu reduzieren. Der Betrag der Beschleunigung r liegt um den Wert 1 herum, wenn sich die Wiimote in Ruhelage befindet. Liegt nun der Beschleunigungsbetrag innerhalb definierter Grenzen um 1 herum, so werden diese Daten durch den eingebauten Filter grundsätzlich als Gestendaten ausgeschlossen. Einen ähnlichen Ansatz verfolgen auch Malmestig und Sundberg in [MS08], bei denen allerdings die einfachen Beschleunigungswerte $> 2g$ sein müssen.
Prekopcsák [Pre08] verwendet einen anderen Ansatz. Er nimmt nicht den Beschleunigungsbetrag des eigentlichen Vektors, sondern H, den Betrag der Änderung (Steigung) zum vorherigen Vektor. Dieser ist größer, wenn eine Richtungsänderung vorliegt und das entspricht definitionsgemäß einem Gesten-Segment. Befindet sich die Wiimote in Ruhelage und ändert ihre Richtung nur sehr gering, dann sollte der Betrag der Änderung gegen null gehen. Prekopcsák hat herausgefunden, dass dabei recht "hektische" Werte herauskommen, so dass er den exponentiellen moving average (EMA) berechnet (siehe 3.3.3). Wenn dieser EMA eine gewisse Grenze überschreitet, nimmt er

an, dass dort eine Geste beginnt. Dieser Ansatz wird später auch im Vergleich getestet.

Eine Alternative scheint es zu sein, vor der Klassifikation nicht explizit zu segmentieren: “Der gewöhnlichste Ansatz im Segmentierungsprozess ist das explizite Extrahieren von kompletten Gesten in einem Strom von Merkmalsvektoren durch irgendeine Form von Aktivierungsfunktion oder Grenzwert. Ein alternativer Ansatz ist es, nicht explizit vor der Klassifizierung zu segmentieren. Diese Alternative ist möglich, wenn ein Zustandsraumansatz, wie HMM, zur Klassifikation verwendet wird. Übergangszustände in einem HMM basieren auf Übergangswahrscheinlichkeiten. Diese Wahrscheinlichkeiten können als Aktivierungsfunktionen gesehen werden. Über diesen Weg segmentiert die Klassifizierungsmethode implizit den Datenstrom durch Verharren in einem Startzustand bis sie einen Gestenstart gefunden hat. Das Ende einer Geste wird gefunden wenn das HMM in einen Endzustand eintritt. Die dritte Möglichkeit ist eine Kombination der beiden genannten Methoden. Diese Methode segmentiert den Datenstrom explizit low level in Gestenteile, anstatt ihn in komplette Gesten zu zerlegen. Diese Gestenteile werden dann in einem HMM kombiniert.“ aus: Hassink et al. [HS06].

3.2. Signalerfassung und Datenaufbereitung

3.2.1. Verwendete Daten

Für die Segmentierung wurden Geste-/Nicht-Geste-Zeitreihen mit einer Dauer von ca. 30 Sek. inklusive eines Gestenindikator aufgezeichnet. Das bedeutet, dass während der Aufzeichnung einer Geste der A-Button der Wiimote gedrückt wurde und somit subjektiv die Geste-Sequenz als solche markiert wurde.

Tab. 3.1.: *Übersicht verwendete Gestendaten mit Button-Gestenindikator*

gestureclass	length	sequence	number	MinLength	MaxLength	MeanLength
circle	34.964s	gesture	12	1.170s	1.440s	1.265s
		non-gesture	13	0.820s	4.510s	1.210s
throw	33.777s	gesture	12	0.430s	0.670s	0.555s
		non-gesture	13	1.450s	2.580s	2.060s
frisbee	33.934s	gesture	15	0.380s	0.660s	0.540s
		non-gesture	16	1.040s	1.930s	1.590s
bowling	34.066s	gesture	13	0.420s	0.580s	0.490s
		non-gesture	14	1.600s	2.600s	1.870s
z	32.379s	gesture	7	1.250s	1.580s	1.370s
		non-gesture	8	1.870s	3.720s	2.605s

gestureclass	sequence	$MinAmpl_r$	$MaxAmpl_r$	$MeanAmpl_r$
circle	gesture	0.19g	2.13g	(1.07±0.46)g
	non-gesture	0.72g	1.49g	(1.02±0.08)g
throw	gesture	0.55g	7.82g	(3.37±2.04)g
	non-gesture	0.59g	6.51	(1.11±0.31)g
frisbee	gesture	0.57g	7.59g	(3.05±2.04)g
	non-gesture	0.46g	2.25g	(1.07±0.22)g
bowling	gesture	0.86g	5.82g	(3.40±1.52)g
	non-gesture	0.18g	2.43g	(1.14±0.30)g
z	gesture	0.80g	2.22g	(1.32±0.27)g
	non-gesture	0.83g	1.20g	(0.98±0.04)g

Es wurde zusätzlich zu den Beschleunigungsdaten acc_x, acc_y, acc_z der Betrag der Beschleunigung r aus diesen errechnet, der in Ruhelage ungefähr den Wert 1 annehmen sollte. Für die Unterscheidung von Gesten- und Ruhezustand der Wiimote wird dieser Betrag r bestehend aus

$$r = \left| \begin{pmatrix} acc_x \\ acc_y \\ acc_z \end{pmatrix} \right| = \sqrt{acc_x^2 + acc_y^2 + acc_z^2} \tag{3.1}$$

verwendet.

In den Abbildungen 3.1 bis 3.5 sind der Beschleunigungsbetrag $r(t)$ und die Beschleunigungswerte $acc_x(t)$, $acc_y(t)$, $acc_z(t)$ über die Zeit der fünf Segmentierungsdatenströme dargestellt. Der grau hinterlegte Bereich entspricht jeweils dem Gesten-Segment, das über den Button markiert wurde. In den nachfolgenden Grafiken sind jeweils die ersten 10 Sekunden der verwendeten Datenströme dargestellt. In den Abbildungen erkennt man, dass die Ruhelagewerte der blauen

Kurve r eher um den Wert 1 angeordnet sind. Diese Tatsache kann man als Indikator für Gestentrennung verwenden, ebenso wie in [HHH98] von Hofmann et al. und [SPHB08] bei Schlömer et al. verwendet. Auch bei der regelbasierten Trennung (vgl. 3.3.1) wird diese Tatsache verwendet.

Verlauf des Beschleunigungsbetrags Die Wurfgesten *throw*, *frisbee* und *bowling* haben alle im Bereich der Geste relativ hohe Peaks und kleinere Nachschwinger. Die ersten beiden mit Werten bis zu 8g, die *bowling*-Geste nur bis zu 6g. Die *throw*-Geste hat zwei Extrempunkte im Peak, erst einen kleineren, dann einen größeren, auf den Peak folgt das Nachschwingen. Die *throw*-Geste hat stärkere Nachschwinger (mit größerer Amplitude) als die *frisbee*-Geste. Die längsten und markantesten Nachschwinger sind allerdings in der *bowling*-Geste zu erkennen.

circle und z weisen hingegen nicht so hohe Werte auf. Der Gestenbereich der *circle*-Geste lässt sich rein optisch im Vergleich zur Ruhelage gut abgrenzen. Er besteht aus einer anderhalbfachen sinusförmigen Schwingung. Bei der z-Geste ist eine Abgrenzung eher schwierig, da das Signal an sich sehr verrauscht ist. Der Gestenbereich besteht aus ansteigenden kleinen Peaks bis 2g. Signale der Würfe sind weniger verrauscht, vermutlich weil dort längere durchgehende Bewegungen ausgeführt wurden.

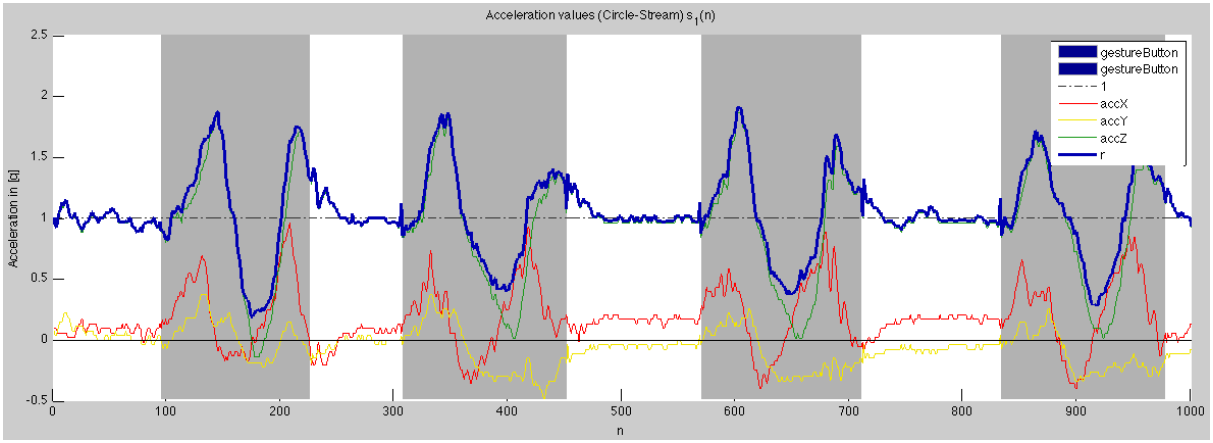

Abb. 3.1.: *Zeitreihe (0-10 Sek.) der circle-Geste mit den aufgezeichneten Beschleunigungsrohdaten $acc_x(t)$, $acc_y(t)$, $acc_z(t)$, dem berechneten Beschleunigungsbetrag r und dem Button-Gestenindikator*

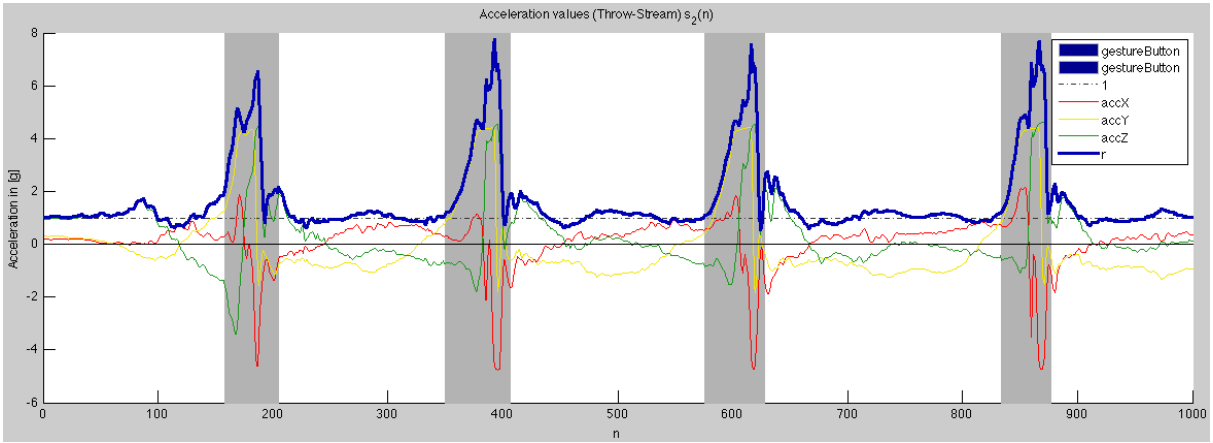

Abb. 3.2.: *Zeitreihe (0-10 Sek.) der throw-Geste mit den aufgezeichneten Beschleunigungsrohdaten $acc_x(t)$, $acc_y(t)$, $acc_z(t)$, dem berechneten Beschleunigungsbetrag r und dem Button-Gestenindikator*

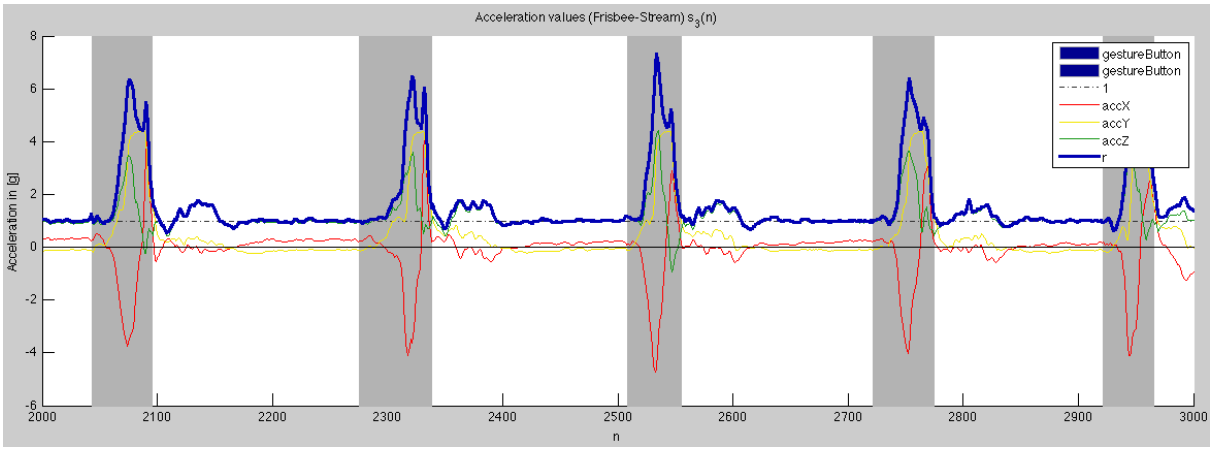

Abb. 3.3.: *Zeitreihe (0-10 Sek.) der frisbee-Geste mit den aufgezeichneten Beschleunigungsrohdaten $acc_x(t)$, $acc_y(t)$, $acc_z(t)$, dem berechneten Beschleunigungsbetrag r und dem Button-Gestenindikator*

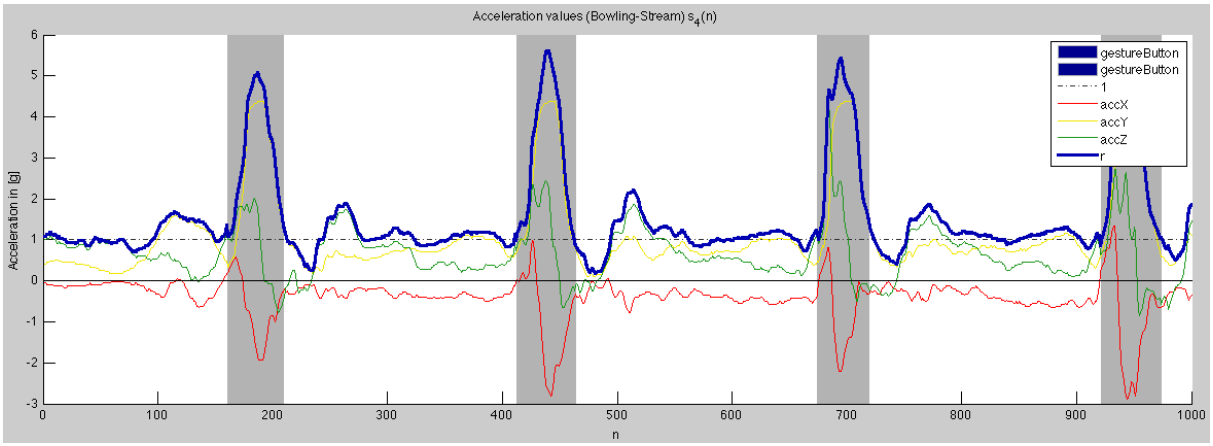

Abb. 3.4.: *Zeitreihe (0-10 Sek.) der bowling-Geste mit den aufgezeichneten Beschleunigungsrohdaten $acc_x(t)$, $acc_y(t)$, $acc_z(t)$, dem berechneten Beschleunigungsbetrag r und dem Button-Gestenindikator*

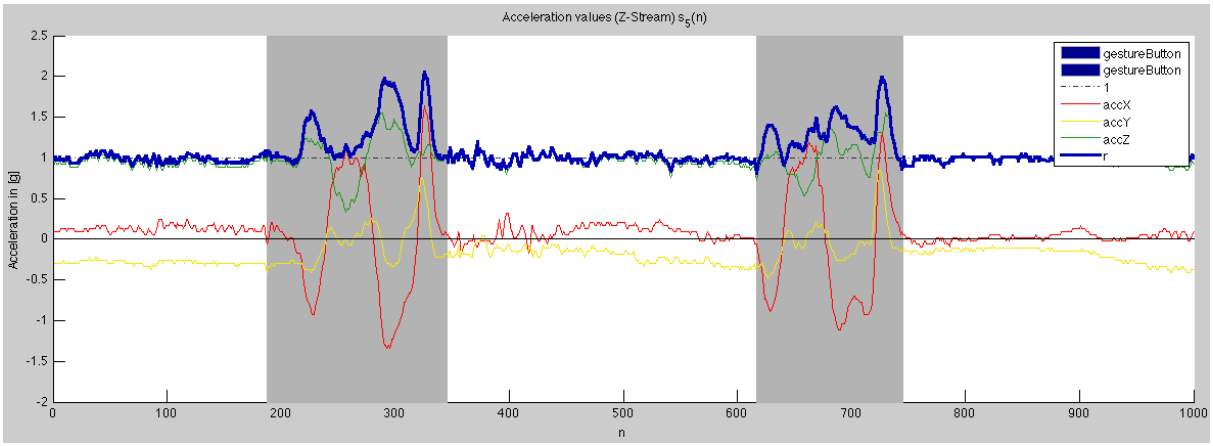

Abb. 3.5.: *Zeitreihe (0-10 Sek.) der z-Geste mit den aufgezeichneten Beschleunigungsrohdaten $acc_x(t)$, $acc_y(t)$, $acc_z(t)$, dem berechneten Beschleunigungsbetrag r und dem Button-Gestenindikator*

3.2.2. Kalibrierung

Bei den Untersuchungen hat sich herausgestellt, dass die Wiimote nicht "richtig" kalibriert ist, d.h. die gemessenen Ruhelagewerte entsprechen nicht den erwarteten Werten. In Tabelle 3.2 sind die gemessenen Werte des verwendeten Geräts exemplarisch dargestellt. Die "direction" zeigt an, welche Seite der Wiimote nach "unten" gerichtet ist. Die Orientierungen der Beschleunigungssensoren der Wiimote wurde bereits in Abb. 2.2 dargestellt. Erwartungsgemäß sollte die nach unten gerichtete Achse den Wert 1 aufweisen, beide anderen sollten 0 sein. Diese Kalibrierungsfehler treten nicht zufällig auf, sondern resultieren aus den Messungenauigkeiten der Beschleunigungssensoren. Sie sind damit für jedes Gerät immer gleich, aber von Gerät zu Gerät unterschiedlich (siehe Tabellen 3.2 und 3.3 im Vergleich).

Tab. 3.2.: *Gemessene Werte des Aufzeichnungsgerätes*

direction	z	-z	-x	x	-y	y
acc_x	0.04	0.04	-0.93	1.04	0.07	0.04
acc_y	0	0	0	0.04	-1	1
acc_z	1.04	-0.92	0.04	0.08	0.08	0.08

Tab. 3.3.: *Gemessene Werte eines nicht verwendeten anderen Gerätes*

direction	z	-z	-x	x	-y	y
acc_x	0	0	-1	1	-0.04	0
acc_y	0	0	0.04	0.04	-1.04	1.04
acc_z	1.04	-0.96	0.04	0.04	0	0

Um die Messwerte den erwarteten Werten anzupassen, ist eine Rekalibrierung implementiert worden, die für dieses Gerät die Werte angleicht. Eine solche Kalibrierung kann nur gerätespezifisch vorgenommen werden. Für die automatische Kalibrierung der Wiimote ist bereits vom Hersteller der Kalibrierungs-Offset des jeweiligen Gerätes in dessen EEProm gespeichert und auslesbar [o.V10], allerdings wird dieser im verwendeten Framework nicht zur automatischen Kalibrierung eingesetzt. Aus diesem Grunde werden die Beschleunigungswerte nachträglich rekalibriert. Für acc_x wäre das exemplarisch $acc_{x\_calib} = a * acc_{x\_orig} + b$, wobei die Koeffizienten a und b durch lösen des folgenden Gleichungssystems aus den gemessenen Werten ermittelt werden:

$$-1 = a(-0.93) + b \tag{3.2}$$

$$+1 = a(1.04) + b \tag{3.3}$$

Analog dazu wird mit den Beschleunigungswerten für y und z verfahren.

Mit dieser Rekalibrierung liegen die Ruhelagewerte besser um 1, was für die regelbasierte Trennung Vorteile bei der Findung der Parameter hat, da sie dann mit gleichem Abstand um 1 zunächst angenommen werden können. Bei allen Gesten außer der z-Geste ergeben sich durch die Rekalibrierung bessere Ergebnisse als ohne (siehe dazu Tabelle 3.4). Unkalibriert weicht r stärker nach unten ab, wie man in der Darstellung 3.6 sieht. Außerdem basiert auch die Ruhelage-Erkennung der Java-Anwendung auf dieser Annahme und lässt durch die Rekalibrierung eine feinere Einstellung des oberen und unteren Grenzwertes zu. Auf die Klassifizierung mit SFA hat die Kalibrierung keinen wesentlichen Einfluss (siehe Tabelle 3.5). In der folgenden Grafik ist der Beschleunigungsbetrag mit und ohne Kalibrierung im Vergleich aufgetragen.

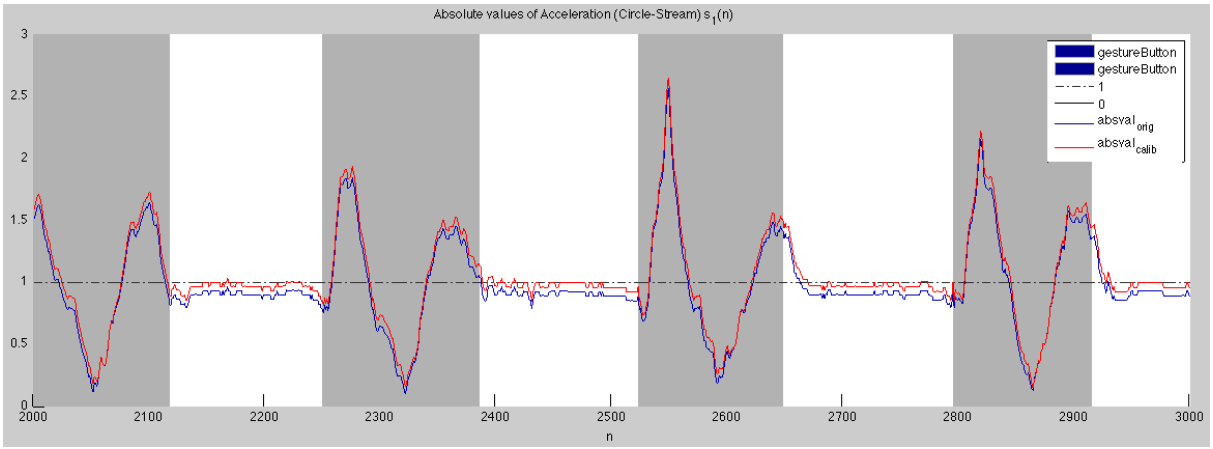

Abb. 3.6.: *Demonstration der Auswirkung der Kalibrierung auf r_{circle} (original=blau und rekalibriert=rot)*

3.3. Segmentierungstests

Um einen möglichen Einsatz der SFA zur Segmentierung zu untersuchen, wurden mehrere Ansätze verfolgt. Grundsätzlich wurden drei Verfahren zur Bestimmung von Anfang und Ende einer Geste untersucht, um sich nicht allein auf die subjektive Bestimmung von Gesten durch den Probanden zu beschränken: eine regelbasierte Segmentierung, die Segmentierung mit der SFA und eine Segmentierung über die durchschnittliche Bewegungsänderung.

Als Maßstab für die Untersuchungen zur Segmentierung wurden die jeweils erkannten Gesten in Relation zu den Gesten gesehen, die bei der Aufzeichnung durch Buttondruck markiert wurden. Dazu wurde bei der Aufzeichnung der Daten immer dann ein Flag gesetzt, wenn der Button gedrückt wurde. Dieses Flag wird im weiteren als Button-Gestenindikator bezeichnet. Für die jeweilig untersuchte Segmentierung wurde ebenfalls ein solcher Gestenindikator kalkuliert. Die Güte der berechneten Gestenindikatoren I_{calc} wurden daran gemessen, ob sie erstens die gleiche Gestenanzahl erkennen und zweitens wie stark sie mit dem Button-Gestenindikator I_{button} korrelieren. Die Abweichung beider Indikatoren Var wird bestimmt durch:

$$Var = \frac{\sum_{n=1}^{N} |I_{button}(n) - I_{calc}(n)|}{N}, \text{ N = Länge der Zeitreihe} \tag{3.4}$$

3.3.1. Regelbasierte Segmentierung

Für die regelbasierte Trennung wird der Beschleunigungsbetrag r verwendet. Wie in Abb. 3.7 dargestellt, werden um den Ruhelagewert 1 zwei Korridore mit $k_{out} = [1-p, 1+p]$ (outerLimits) und $k_{in} = [1-q, 1+q]$(innerLimits) gelegt, wobei $p > q$ ist. Die genauen Parameter für p und q sind zu bestimmen. Betrachtet wird nun r über die Zeit t, wobei r zu Beginn innerhalb des großen Korridors $r \in k_{out}$ sei. Sobald r diesen Bereich verlässt ($r(t) \notin k_{out}$), werden rückwärtig die vorherigen Betragswerte $r(t-n)$ betrachtet, die bereits außerhalb des kleinen Korridors lagen, $r(t-n) \notin k_{in}$. Der erste Austrittspunkt aus dem kleinen Korridor wird als potenzieller Gestenanfang markiert. Es wird nun r weiter über die Zeit verfolgt. Erst sobald r wieder in den kleinen Korridor eintritt, $r(t) \in k_{in}$, wird dieser Eintrittspunkt als Gestenende angenommen(Amplituden-Hysterese). Verlässt r innerhalb eines zu definierenden Zeitraums len_{ng} wieder den kleinen Korridor, so wird angenommen, dass die Geste noch nicht beendet war und es wird das nächste Gestenende gesucht. Bleibt r länger als len_{ng} innerhalb des kleinen Korridors (Zeit-Hysterese), so gilt die vorherige Geste damit als beendet und das System befindet sich im Nicht-Gesten-Zustand. Wird nun der große Korridor wieder verlassen, so beginnt der Prozess von vorne. Um sowohl zu kurze Nicht-Gesten-Bereiche, als auch zu kurze Gesten auszuschließen, müssen zusätzlich zu den Parametern p und q noch Mindestlängen für Geste len_g und Nicht-Geste len_{ng} definiert werden.

Die passenden Parameter p und q sowie die Mindestlängen von Geste len_g und Nicht-Geste len_{ng} müssen dabei für jede Gesten-Zeitreihe neu bestimmt werden.
Experimentell wurden folgende Werte als optimale Parameter für die regelbasierte Trennung ermittelt. Diese sind nun speziell auf die einzelnen rekalibrierten Gestenstreams ausgelegt. Die Fehlerrate entspricht in diesem Fall jeweils der Abweichung zu den über Button markierten Geste/Nicht-Geste-Segmenten. Var_c gibt die Abweichung zum Button-Gestenindikator für die kalibrierten, Var_{nc} die Abweichung für die nicht kalibrierten Daten an.

Die so ermittelte Mindestlänge für Geste und Nicht-Geste liegen nahe der jeweils kürzesten Gesten-/Nicht-Gestenlänge der einzelnen Streams (siehe Tabelle 3.1).

Tab. 3.4.: *Optimale Parametereinstellungen für regelbasierte Segmentierung (Hysterese)*

gesture	$MIN(len_g)$	$MIN(len_{ng})$	p	q	Var_c	Var_{nc}
Circle	1.15 sec	0.90 sec	0.33	0.17	6.21%	6.72%
Throw	0.36 sec	1.50 sec	0.98	0.67	8.08%	9.00%
Frisbee	0.32 sec	1.10 sec	0.91	0.14	8.84%	10.93%
Bowling	0.46 sec	1.20 sec	1.43	0.17	2.47%	4.11%
Z	1.21 sec	2.00 sec	0.21	0.10	3.55%	3.21%

Exemplarisch ist diese regelbasierte Segmentierung in Abb. 3.7 dargestellt, alle weiteren grafischen Darstellungen der Ergebnisse befinden sich im Anhang A.3. Die blauen Marker zeigen an, wann regelbasiert eine Geste erkannt wurde. Die grünen Punkte entsprechen der Markierung über den gedrückten Wiimote-Button. Die grauen Flächen markieren die Gestenbereiche (hellgrau = Übereinstimmung von regelbasiertem Gestenindikator(RB) mit Button-Gestenindikator, dunkelgrau = Abweichungen der beiden Gestenindikatoren). Wie man an den Abweichungen Var in Tabelle 3.4 sieht, ist diese regelbasierte Segmentierung für einige Gesten besser geeignet als für andere.

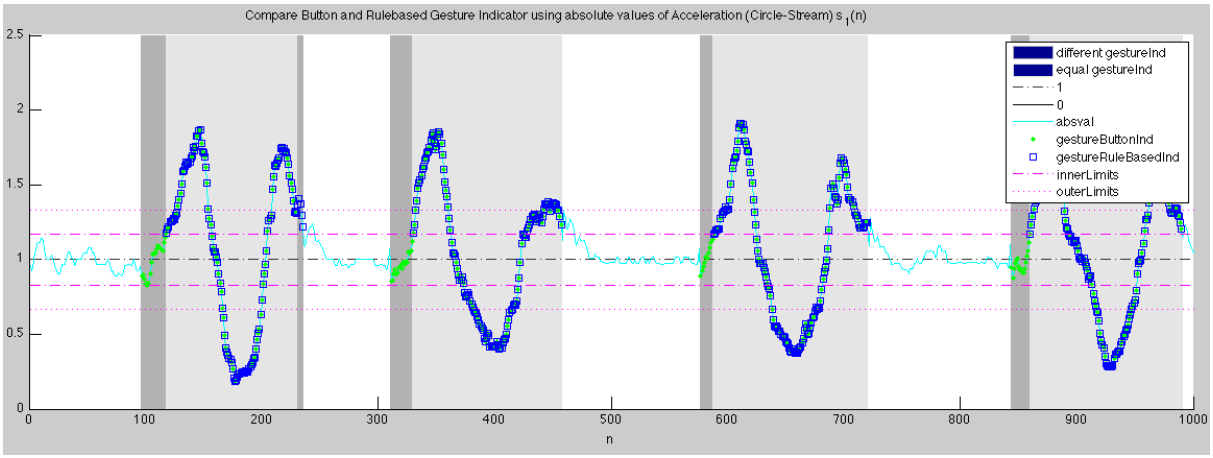

Abb. 3.7.: *Regelbasierte Segmentierung der circle-Gestenzeitreihen (0-10 Sek.) unter Verwendung von zwei Grenzbereichen p(outerLimits) und q(innerLimits) um den Beschleunigungsbetrag r und mit grau hinterlegten Gestenindikatoren (hellgrau = Übereinstimmung von RB- und Button-Gestenindikator, dunkelgrau = abweichende Erkennung einer Geste von einem der beiden Gestenindikatoren). Grüne Marker markieren zusätzlich noch den Bereich in dem der Button-Gestenindikator eine Geste anzeigt, die blauen Marker zeigen die Bereiche, in denen der RB-Gestenindikator eine Geste bestimmt hat.*

3.3.2. Segmentierung mit SFA

Zur Segmentierung mit der SFA wurden drei Ansätze verfolgt:

1. Einfache SFA (2.Grades): als Eingangssignal wurde der Beschleunigungsbetrag r verwendet.

2. Einfache SFA (2.Grades): als Eingangssignal wurden die Beschleunigungsrohdaten acc_x, acc_y und acc_z verwendet und die SFA einmal angewendet.

3. Kaskadierte SFA (2.Grades): als Eingangsdaten für die erste Ausführung der SFA wurden die Beschleunigungsrohdaten acc_x, acc_y und acc_z verwendet und auf den ersten SFA-

Output nochmals die SFA angewendet, um ein größeres Set an Funktionen und nicht nur die Monome 2.Grades zu verwenden.

Für die Untersuchungen wurde aus dem SFA-TK [Kon09] das *drive1*-Skript verwendet, das zur Detektion von Driving Force wie von Wiskott [Wis03] beschrieben verwendet wird. Die Eingangssignale für die SFA wurden jeweils mit unterschiedlichen *embedding*-Dimensionen getestet. Die *embedding*-Dimension $m \in \mathbb{N}$ entspricht der Größe des jeweils betrachten Fensters des Eingangssignals. Sie ist äquivalent zu der reduzierten Dimension, die durch die PCA bei der Klassifizierung entsteht. Durch diese Fenster ist das Ausgangssignal der SFA um m kürzer als das Eingangssignal und wurde deswegen für die weitere Verarbeitung um $\frac{m}{2}$ vorne und $\frac{m}{2}$ hinten mit Nullen aufgefüllt, d.h. zu jedem Zeitpunkt des Ausgangssignals werden die $\frac{m}{2}$ Eingangswerte davor und dahinter betrachtet.

Da für unterschiedliche m die Orientierung des SFA-Outputs zufällig wechselt, weil die Ausrichtung der Eigenwerte nicht einheitlich ist, werden in den folgenden Grafiken die Kurven teilweise an der Abszisse gespiegelt.

Einfache SFA mit Beschleunigungsbetrag

Bei einer ersten Betrachtung zur Segmentierung mit SFA wurde als Eingangssignal für die SFA der Beschleunigungsbetrag r verwendet. Zur Veranschaulichung ist der langsamste SFA-Output y_1 für die *embedding*-Dimensionen $m = \{5, 10, 15, 20, 25, 30, 35\}$ in Abb. 3.8 dargestellt (Darstellung für alle fünf Gesten siehe A.4). Die grauen Flächen markieren dabei jeweils den Gesten-Bereich des Button-Gestenindikators. Der SFA-Output y_1 zeigt in Gestenbereichen stärkere Ausschläge. Die Maximalwerte (Peaks) liegen auch für unterschiedliche m übereinander. In Ruhephasen befinden sich die Werte abhängig von m um eine gewisse “Grundlinie"herum.

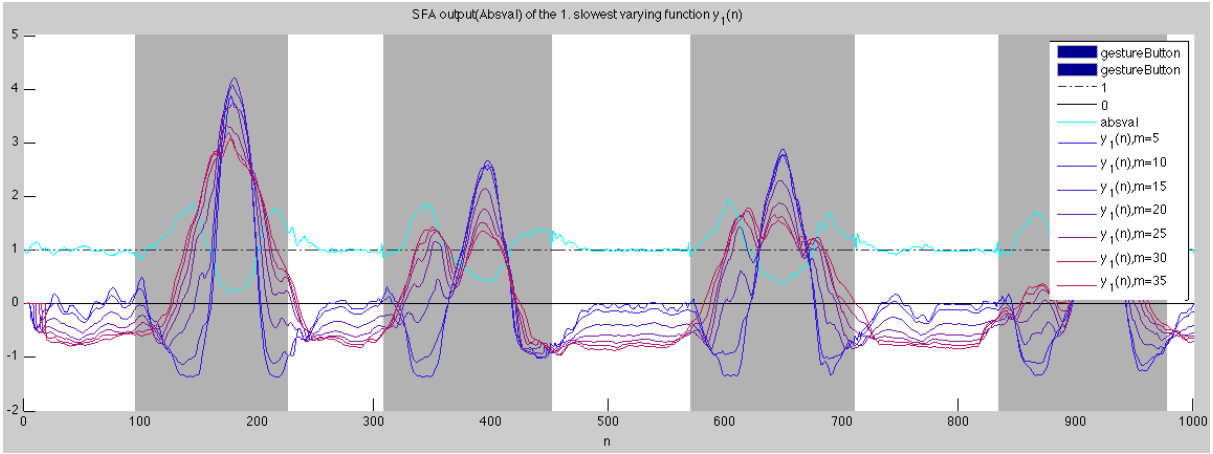

Abb. 3.8.: *Segmentierung mit SFA: Beschleunigungsbetrag r_{Circle} als Input und SFA Output (embedding-dim:5-35)*

Einfach SFA mit Beschleunigungsrohdaten

Als Eingangssignal werden die Beschleunigungsrohdaten verwendet. Die dreidimensionalen Beschleunigungsvektoren einer Zeitreihe werden dazu folgendermaßen konkateniert:

$$sfa_{input} = [acc_x(1), acc_y(1), acc_z(1), acc_x(2), acc_y(2), ...] \tag{3.5}$$

Das daraus resultierende eindimensionale Signal wird nun mit den *embedding*-Dimensionen $m = \{15, 30, 45, 60, 75\}$ als Eingangssignal für die SFA verwendet. Wie in Abb. 3.9 zu sehen, sind qualitativ die Ergebnisse sehr ähnlich zu denen, wenn der "vorverarbeitete" Beschleunigungsbetrag verwendet wird, da der Beschleunigungsbetrag indirekt fast in dem expandierten Vektor der SFA enthalten ist (es werden bei der Expansion u.a. die Monome x^2, y^2, und z^2 gebildet, die auch Elemente bei der Berechnung des Beschleunigungsbetrages sind). Die Darstellung aller fünf Gesten befindet sich in Anhang A.5.

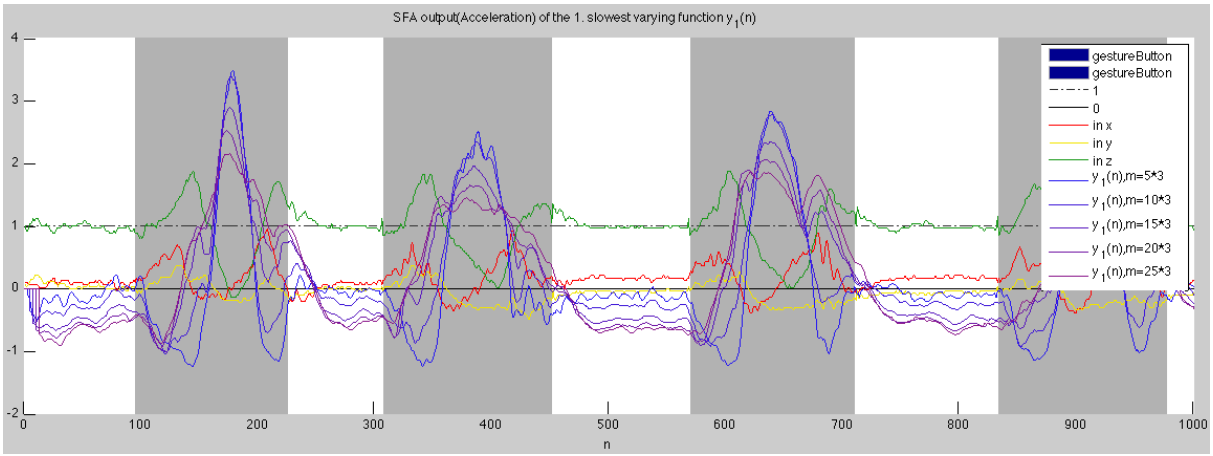

Abb. 3.9.: *Segmentierung mit SFA (Vorüberlegung): Beschleunigungsrohdaten acc_{Circle} als Input und SFA Output (embedding-dim:15-75)*

Erkennbar wird, dass die exakten Werte des SFA-Outputs weniger relevant sind, als die allgemeine Signaleigenschaft des SFA-Outputs. Je höher m gewählt wird, desto stärker wird das Signal "geglättet". Damit werden die Abweichungen vom Median höher und das macht eine einfache Bestimmung von Gesten-Segmentierungseigenschaften für größere m ungenauer. Aus diesem Grund wird für die spätere Segmentierung mit SFA ein relativ kleines m, $m = 15$ gewählt.

Zur genaueren Betrachtung, wie eine Segmentierung möglich ist, wurden einige statistische Werte des Signal betrachtet. Es hat sich herausgestellt, dass der Median am ehesten an der "Grundlinie" liegt, weswegen eine Segmentierung über den Median näher betrachtet wurde. In der folgenden Abbildung sind exemplarisch für die kalibrierte *circle*-Geste jeweils wieder das langsamste von der SFA gefundene Signal y_1 der *embedding*-Dimensionen 15-75 dargestellt. Zusätzlich wurde der Median jedes SFA-Ausgangssignal, der zur Segmentierung der SFA verwendet wird, geplottet.

In der Tabelle 3.5 sind nochmals der Median *median* für kalibrierte und nicht-kalibrierte Daten für die jeweilige *embedding*-Dimension m dieses Versuchs dargestellt. Die Dimensionen wurden im 3er-Abstand gewählt, weil bei der Untersuchung die einzelnen Beschleunigungsvektoren kontakteniert wurden und diese dreidimensional sind. Die Gegenüberstellung von den Ergebnissen der kalibrierten und nicht-kalibrierten Daten zeigt, dass die Re-Kalibrierung auf die Ergebnis der SFA fast keine Auswirkung hat. Das Einzige, was sich durch die Rekalibrierung manchmal ändert, ist die "Orientierung" der Eigenwerte (erkennbar daran, dass der Median mal positiv und mal negativ ist).

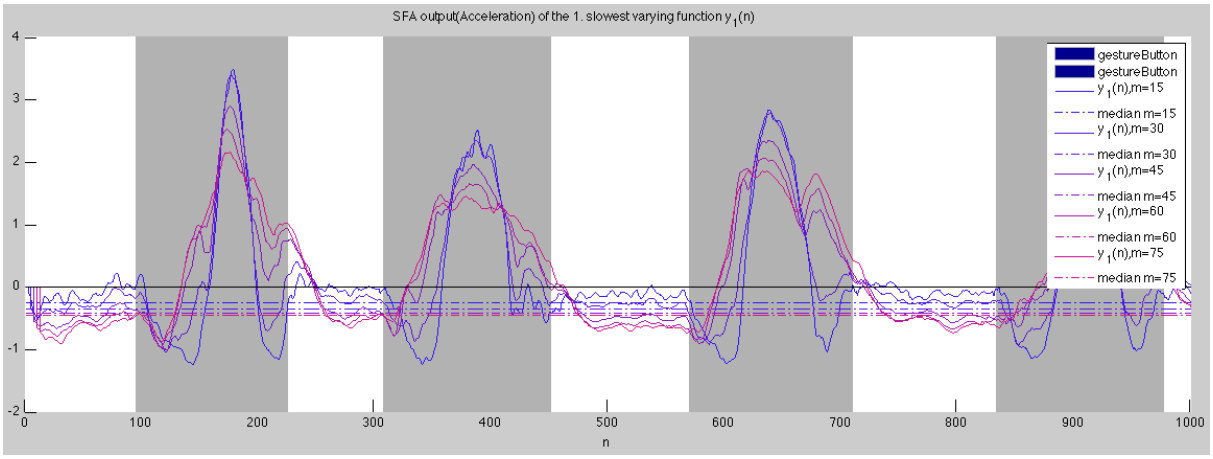

Abb. 3.10.: *Segmentierung mit einfacher SFA über Beschleunigungsrohdaten* acc_{circle_c}*: 1.slowest (embedding-dim:15-75) +* $Median_c$ *(mirrored)*

Tab. 3.5.: *Median für kalibrierte und nicht-kalibrierte Eingangsdaten des langsamsten Signals (das von der SFA gefunden wurde) exemplarisch für den Circle-Stream über* $m = \{3 - 75\}$

m	$median(y_1)kalibriert$	$median(y_1)nichtkalibriert$
3	0.2517	0.2527
6	-0.2499	-0.2481
9	-0.2359	-0.2352
12	0.2417	0.2411
15	0.2517	-0.2513
18	0.2684	0.2678
21	-0.2918	0.2912
24	0.3183	-0.3169
27	0.3419	-0.3411
30	0.3553	0.3536
33	0.3655	0.3612
36	0.3723	-0.3681
39	0.3947	-0.3899
42	-0.4053	-0.3982
45	-0.4210	0.4132
48	-0.4344	0.4281
51	-0.4448	0.4370
54	0.4549	0.4485
57	-0.4552	0.4504
60	0.4608	-0.4538
63	-0.4672	-0.4584
66	-0.4682	0.4602
69	-0.4660	-0.4579
72	-0.4621	-0.4497
75	0.4498	-0.4376

Kaskadierte SFA mit Beschleunigungsrohdaten

Um nicht nur ein Funktionsset der SFA 2.Grades zu Verfügung zu haben, wurde überlegt die SFA zu kaskadieren und sie zweimal hintereinander auszuführen.
Für die erste Durchführung der SFA werden als Eingangssignal die Beschleunigungsrohdaten acc_x, acc_y und acc_z mit der *embedding*-Dimension $m = \{15\}$ verwendet. Der langsamste SFA-Output y_1 wird als neues Eingangssignal für eine zweite Durchführung der SFA verwendet. In Abb. 3.11 ist die zweite Durchführung der SFA mit $m = \{15, 30, 45, 60\}$ dargestellt. Die cyanfarbende Kurve des ersten SFA-Outputs bzw. Inputs für die zweite SFA ist in dieser Darstellung nicht gespiegelt, entspricht also der originalen Orientierung von y_1. Die Ergebnisse der zweiten SFA sind "glatter" als die der ersten.

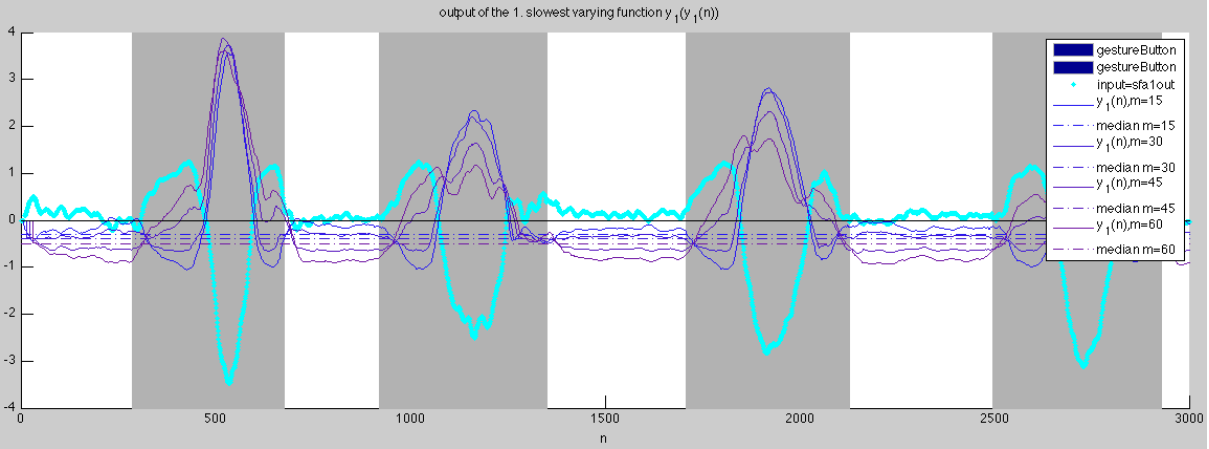

Abb. 3.11.: *Segmentierung mit kaskadierter SFA über Beschleunigungsrohdaten: Beschleunigungsrohdaten acc_{circle} als 1.Input mit $m = 15$ und SFA Output y_1 als zweiten Input mit (embedding-dim:15-60)*

SFA-Segmentierung über Grenzen um Median

Für alle drei Ansätze zur Segmentierung mit der SFA wurden zwei Grenzen mit gleichem Abstand um den Median des SFA-Outputs y_1 gelegt, $median(y_1) + p$ und $median(y_1) - p$. Liegt das Signal außerhalb dieser Grenzen, wird eine Geste angenommen, liegt es innerhalb, wird davon ausgegangen, dass es sich um Ruhelage handelt. Zusätzlich wurden ebenfalls wieder zu kurze Gesten bzw. Nicht-Gesten ausgeschlossen. Da keine einheitlichen Werte für die Grenze p, die Mindestlänge der Geste $MIN(len_g)$ und die Mindestlänge des Nicht-Gesten-Segments $MIN(len_{ng})$ über alle Gesten bestimmt werden konnten, wurden experimentell für jede Geste die optimalen Parameter einzeln ermittelt. Als optimale Parametereinstellungen wurden die Werte ermittelt, bei denen die geringsten Abweichungen zum Button-Gestenindikator besteht. Die so gefundenen Parametereinstellungen für die drei SFA Ansätze sind in den folgenden Tabellen dargestellt sowie jeweils die Abweichung Var zum Button-Gestenindikator.

Tab. 3.6.: *Optimale Parametereinstellungen für Segmentierung bei Verwendung von einfacher SFA ($m = 10$) mit Beschleunigungsbetrag r und Median(y_1)*

gesture	$MIN(len_g)$	$MIN(len_{ng})$	p	Var
circle	1.10 sec	0.98 sec	0.44	6.73%
throw	0.36 sec	1.56 sec	1.20	7.36%
frisbee	0.30 sec	0.36 sec	0.83	10.40%
bowling	0.36 sec	0.15 sec	0.83	4.15%
z	1.16 sec	0.57 sec	0.95	3.26%

Tab. 3.7.: *Optimale Parametereinstellungen für Segmentierung bei Verwendung von einfacher SFA ($m = 15$) mit Beschleunigungsrohdaten acc und Median(y_1)*

gesture	$MIN(len_g)$	$MIN(len_{ng})$	p	Var
circle	0.91 sec	0.76 sec	0.37	6.47%
throw	0.34 sec	0.30 sec	0.81	6.10%
frisbee	0.27 sec	0.32 sec	0.47	11.24%
bowling	0.32 sec	0.17 sec	0.72	5.28%
z	1.09 sec	1.60 sec	1.46	3.80%

Tab. 3.8.: *Optimale Parametereinstellungen für Segmentierung bei Verwendung von kaskadierter SFA ($m_1 = 15$, $m_2 = 15$) mit Beschleunigungsrohdaten acc und Median(y_1)*

gesture	$MIN(len_g)$	$MIN(len_{ng})$	p	Var
circle	0.95 sec	0.95 sec	0.51	6.97%
throw	0.23 sec	0.30 sec	1.10	6.82%
frisbee	0.25 sec	0.31 sec	0.79	11.42%
bowling	0.35 sec	0.16 sec	0.80	4.74%
z	1.10 sec	1.60 sec	1.33	3.19%

Die Ergebnisse der Segmentierung mit den ermittelten Parametern der drei Ansätze sind im folgenden für alle fünf Gesten grafischen gegenübergestellt.

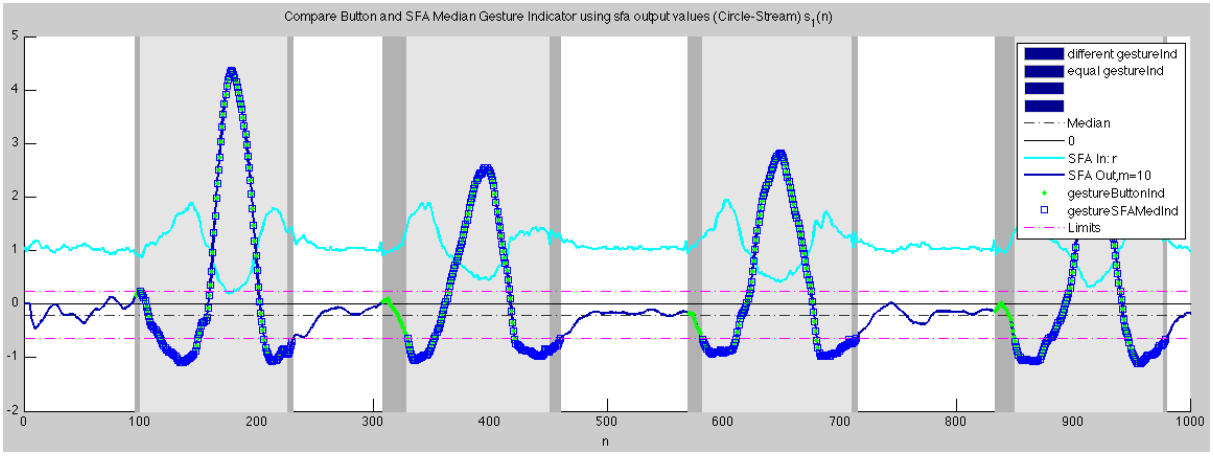

Abb. 3.12.: *Vergleich Segmentierung mit SFA über Beschleunigungsbetrag (circle-Geste): Beschleunigungsbetrag r_{Circle} als Input und SFA Output $y_1(m = 10)$*

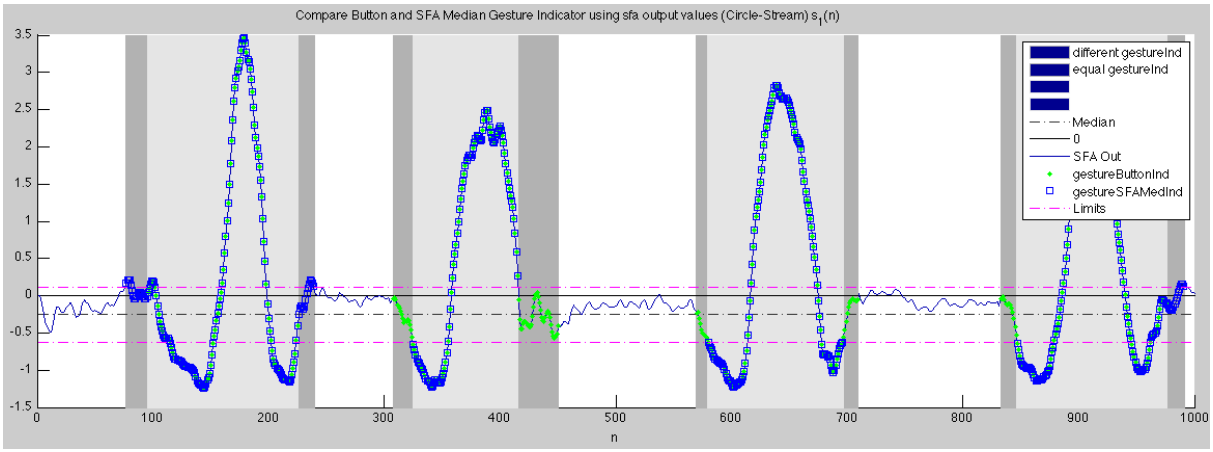

Abb. 3.13.: *Vergleich Segmentierung mit einfacher SFA über Beschleunigungsrohdaten (circle-Geste): Beschleunigungsrohdaten acc_{Circle} als Input und SFA Output y_1(m=15)*

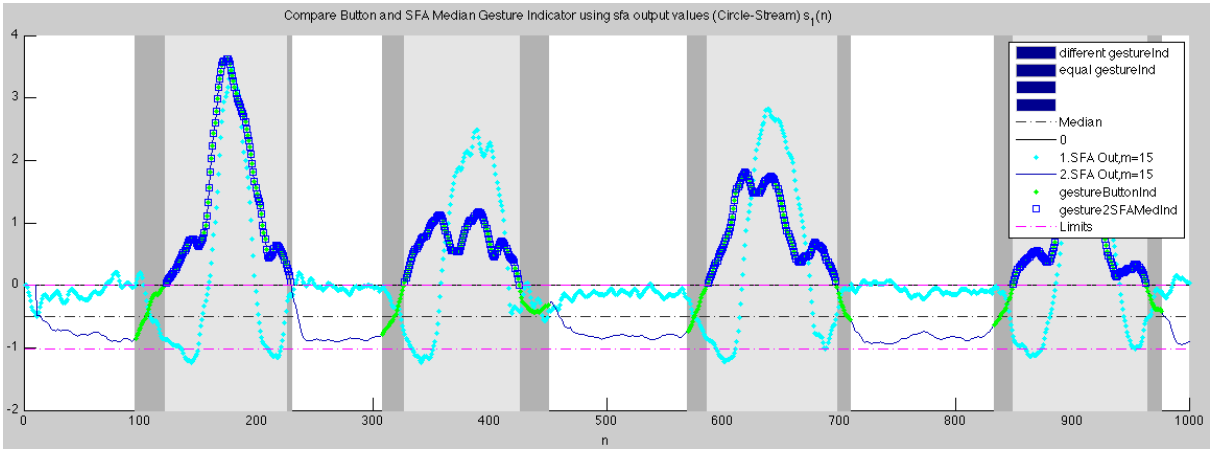

Abb. 3.14.: *Vergleich Segmentierung mit kaskadierter SFA über Beschleunigungsrohdaten (circle-Geste): Beschleunigungsrohdaten acc_{Circle} als 1.Input mit $m = 15$ und SFA Output y_1 als zweiten Input mit $m = 15$*

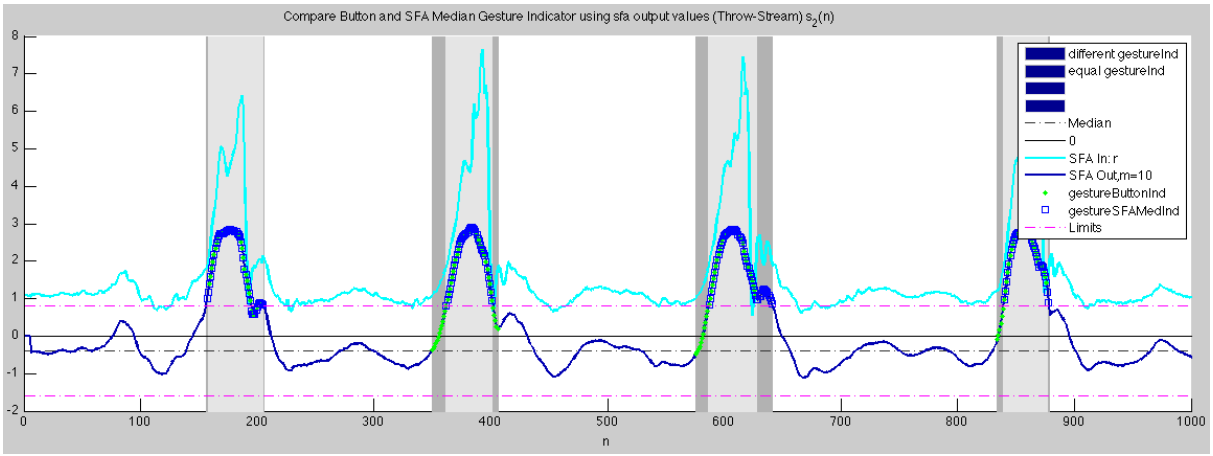

Abb. 3.15.: *Vergleich Segmentierung mit SFA über Beschleunigungsbetrag (throw-Geste): Beschleunigungsbetrag r_{throw} als Input und SFA Output $y_1(m = 10)$*

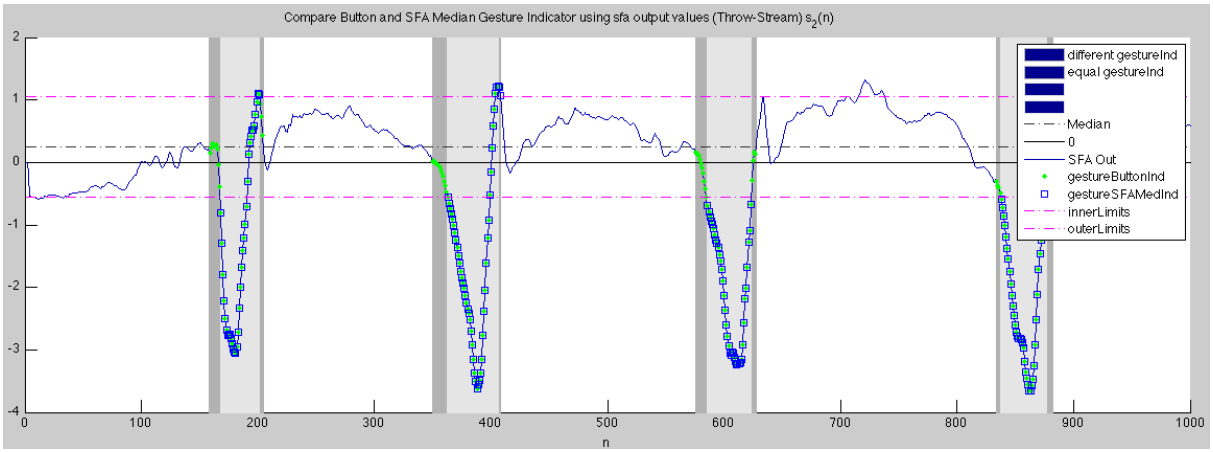

Abb. 3.16.: *Vergleich Segmentierung mit einfacher SFA über Beschleunigungsrohdaten (throw-Geste): Beschleunigungsrohdaten acc_{throw} als Input und SFA Output $y_1(m{=}15)$*

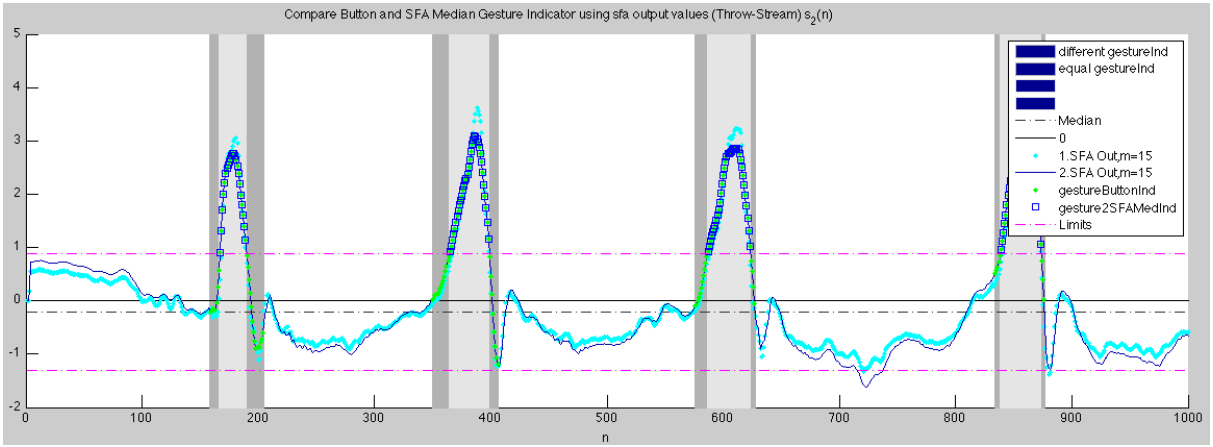

Abb. 3.17.: *Vergleich Segmentierung mit kaskadierter SFA über Beschleunigungsrohdaten (throw-Geste): Beschleunigungsrohdaten acc_{throw} als 1.Input mit $m = 15$ und SFA Output y_1 als zweiten Input mit $m = 15$*

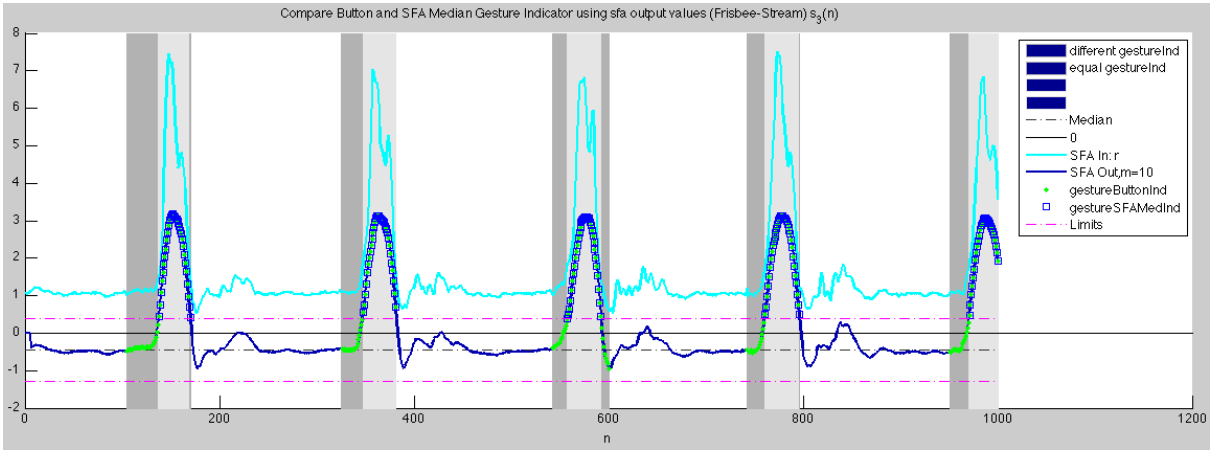

Abb. 3.18.: *Vergleich Segmentierung mit SFA über Beschleunigungsbetrag (frisbee-Geste): Beschleunigungsbetrag $r_{frisbee}$ als Input und SFA Output $y_1(m = 10)$*

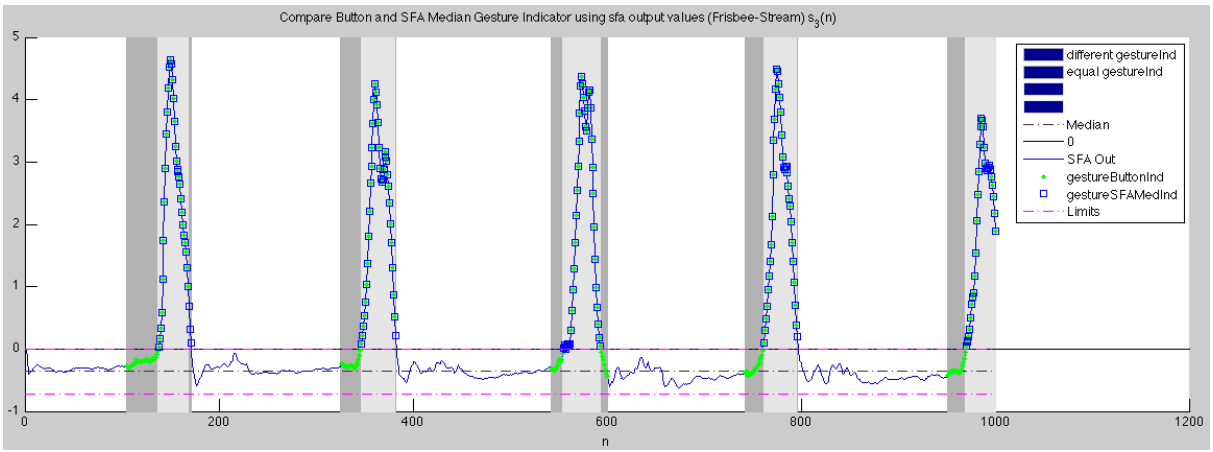

Abb. 3.19.: *Vergleich Segmentierung mit einfacher SFA über Beschleunigungsrohdaten (frisbee-Geste): Beschleunigungsrohdaten $acc_{frisbee}$ als Input und SFA Output $y_1(m{=}15)$*

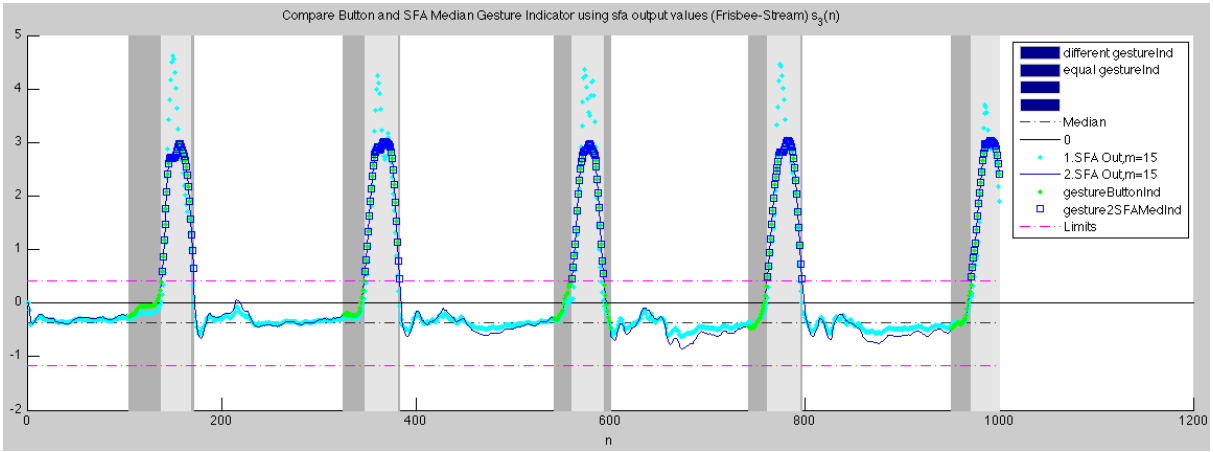

Abb. 3.20.: *Vergleich Segmentierung mit kaskadierter SFA über Beschleunigungsrohdaten (frisbee-Geste): Beschleunigungsrohdaten $acc_{frisbee}$ als 1.Input mit $m = 15$ und SFA Output y_1 als zweiten Input mit $m = 15$*

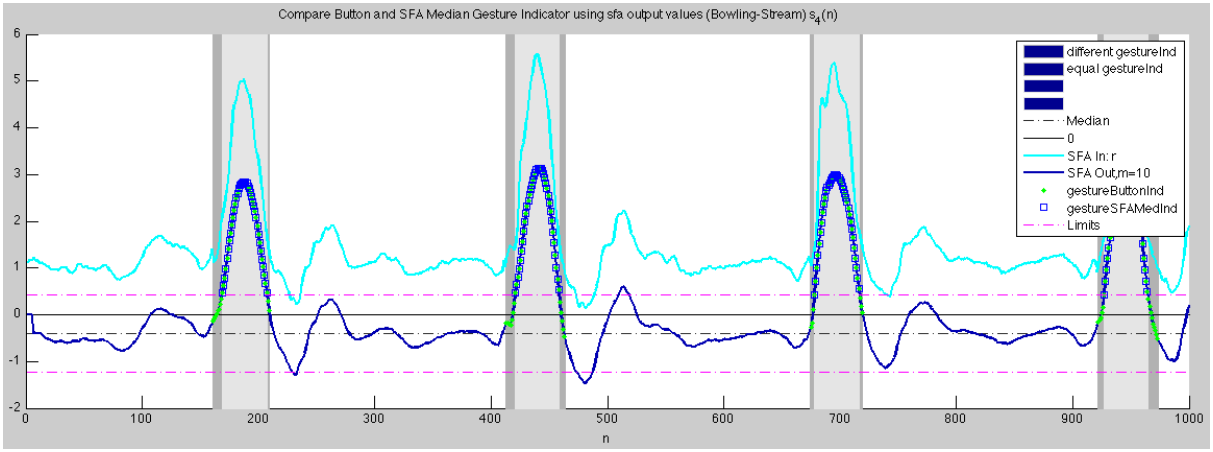

Abb. 3.21.: *Vergleich Segmentierung mit SFA über Beschleunigungsbetrag (bowling-Geste): Beschleunigungsbetrag $r_{bowling}$ als Input und SFA Output $y_1(m = 10)$*

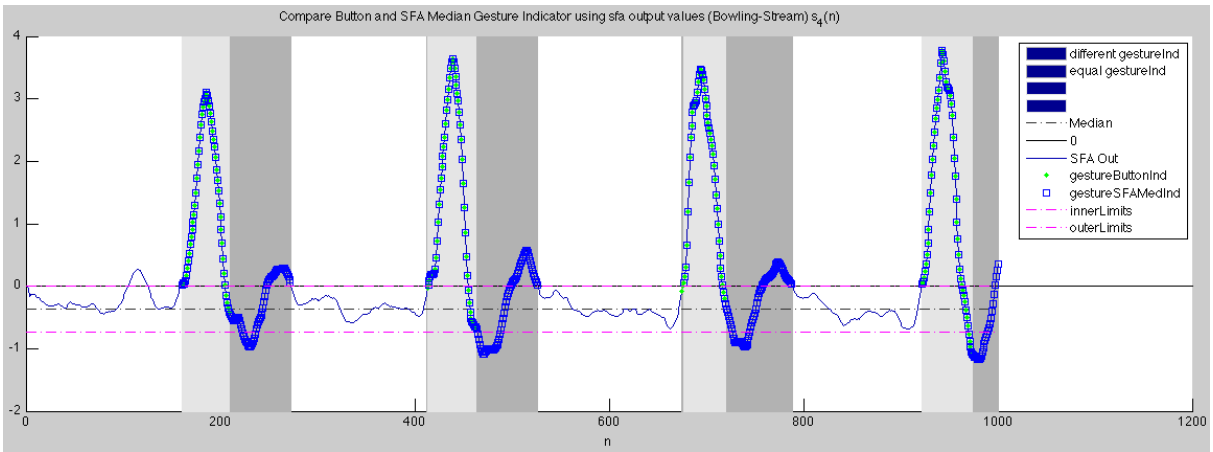

Abb. 3.22.: *Vergleich Segmentierung mit einfacher SFA über Beschleunigungsrohdaten (bowling-Geste): Beschleunigungsrohdaten $acc_{bowling}$ als Input und SFA Output $y_1(m{=}15)$*

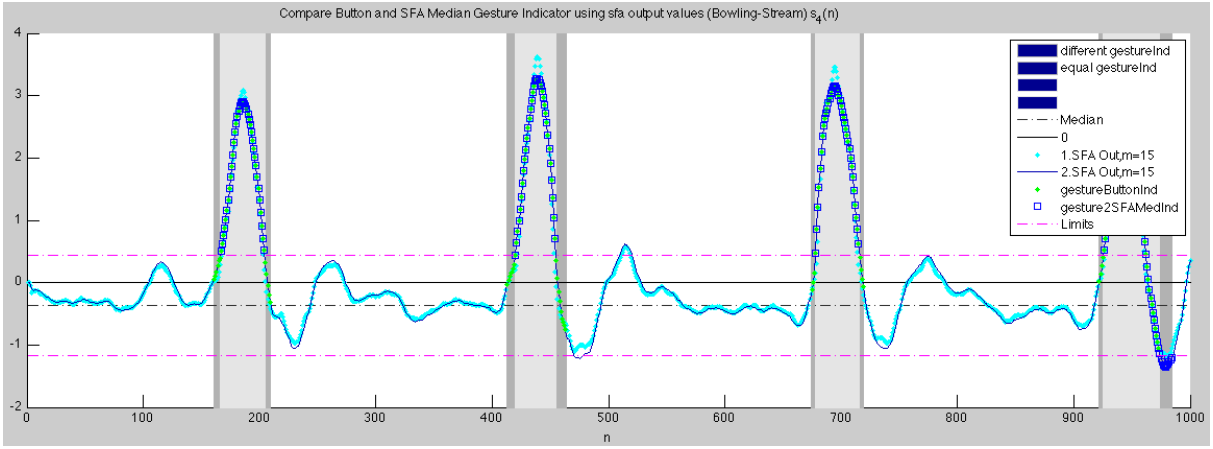

Abb. 3.23.: *Vergleich Segmentierung mit kaskadierter SFA über Beschleunigungsrohdaten (bowling-Geste): Beschleunigungsrohdaten $acc_{bowling}$ als 1.Input mit $m = 15$ und SFA Output y_1 als zweiten Input mit $m = 15$*

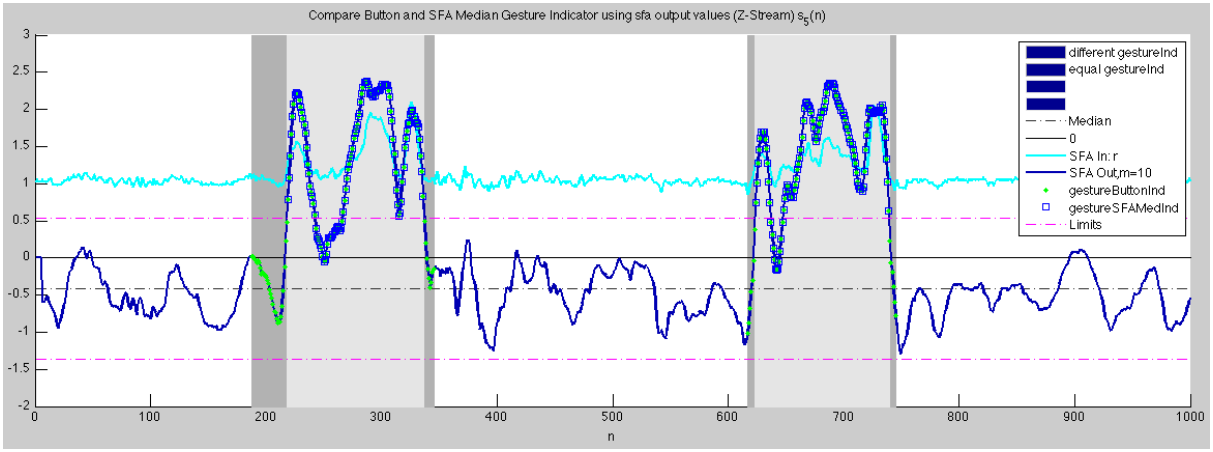

Abb. 3.24.: *Vergleich Segmentierung mit SFA über Beschleunigungsbetrag (z-Geste): Beschleunigungsbetrag r_z als Input und SFA Output $y_1(m = 10)$*

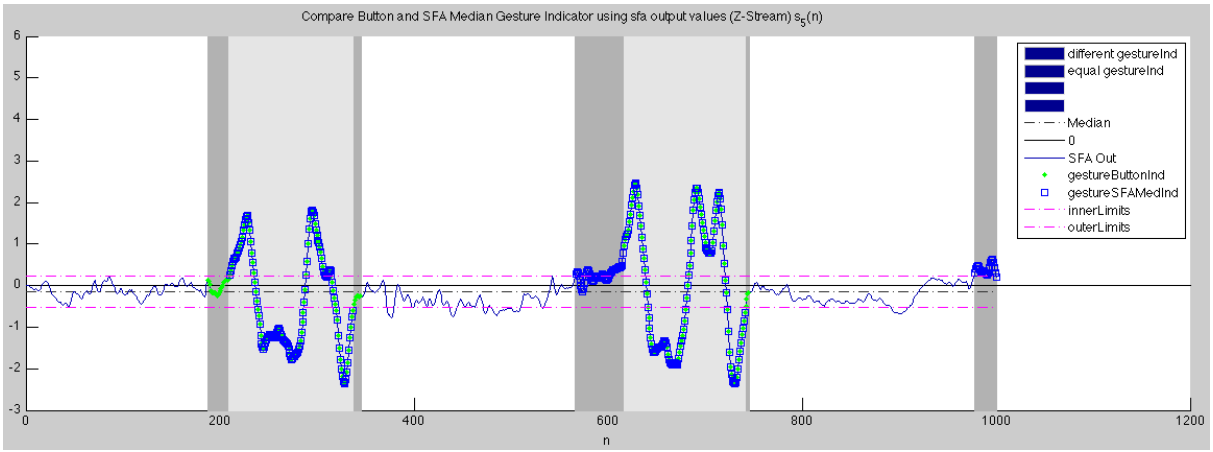

Abb. 3.25.: *Vergleich Segmentierung mit einfacher SFA über Beschleunigungsrohdaten (z-Geste): Beschleunigungsrohdaten acc_z als Input und SFA Output $y_1(m{=}15)$*

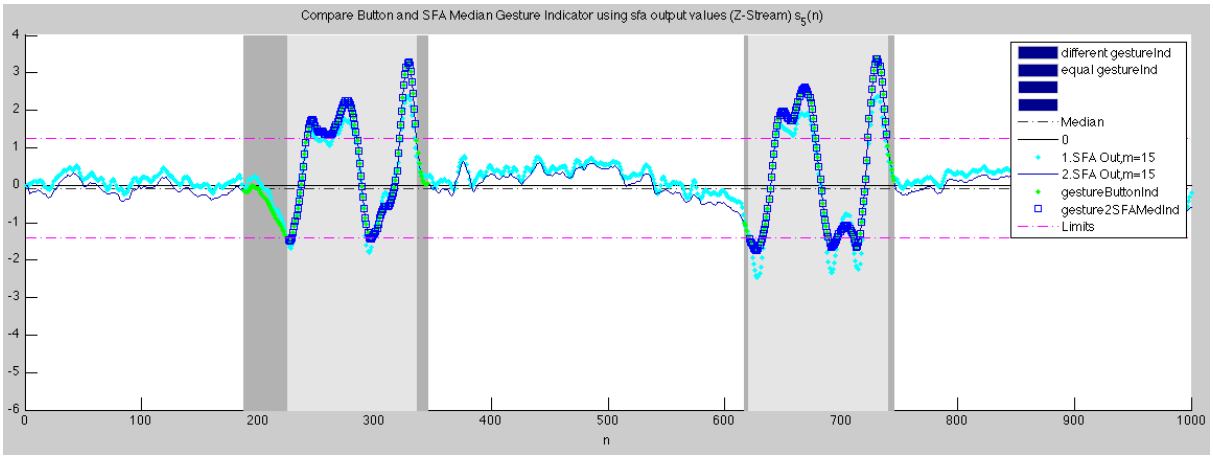

Abb. 3.26.: *Vergleich Segmentierung mit kaskadierter SFA über Beschleunigungsrohdaten (z-Geste): Beschleunigungsrohdaten acc_z als 1.Input mit $m = 15$ und SFA Output y_1 als zweiten Input mit $m = 15$*

3.3.3. Segmentierung über durchschnittliche Bewegungsänderung

Prekopcsák [Pre08] schlägt ein Segmentierungsverfahren vor, mit dem er dynamisch Gesten aufzeichnen kann und das natürlich(er) in der Handhabung sein soll, ohne dass zur Markierung von Gesten ein Button gedrückt werden muss.
Wie bereits in 3.1 beschrieben, wird dazu die Beschleunigungsänderung (Betrag der Steigungen) verwendet. Dieser Steigungsbetrag, $H_k = \sqrt{(x_k - x_{k-1})^2 + (y_k - y_{k-1})^2 + (z_k - z_{k-1})^2}$, für die bereits aufgezeichneten 5 Segmentierungsdatensätze ist in den Abbildungen 3.27 bis 3.31 als rote Kurve dargestellt. Aufgrund der "Unruhe" von H_k wird statt diese zu verwenden, die durchschnittliche Bewegung exponentiell über die Zeit betrachtet:

$$EMA_{H_k} = \alpha H_k + (1 - \alpha) EMA_{H_{k-1}} (\text{EMA} = \text{exponential moving average}) \tag{3.6}$$

Durch EMA_{H_k} wird dann ein Grenzwert gelegt. Liegen die Werte oberhalb dieses Grenzwertes, wird ein Gestenzustand angenommen, unterhalb bedeutet Nicht-Geste. Zusätzlich wird eine Mindestlänge (0.8 sec bei [Pre08], hier siehe Tabelle 3.9) von Gesten festgelegt, um nicht zu kurze Gestensegmente zu finden. α wurde wie bei Prekopcsák mit 0.2 angenommen. Die Grenzwerte wurden für jede Geste wieder separat ermittelt.

Tab. 3.9.: *Optimale Parametereinstellungen für Trennung über durchschnittliche Bewegungsänderung*

gesture	$MIN(len_g)$	$MIN(len_{ng})$	threshold	Var
circle	0.40 sec	0.10 sec	0.04	7.25%
throw	0.40 sec	0.10 sec	0.10	15.42%
frisbee	0.40 sec	0.10 sec	0.15	16.79%
bowling	0.40 sec	0.10 sec	0.12	8.14%
z	0.40 sec	0.10 sec	0.05	7.25%

Die grafischen Resultate sind in den folgenden Abbildungen dargestellt.

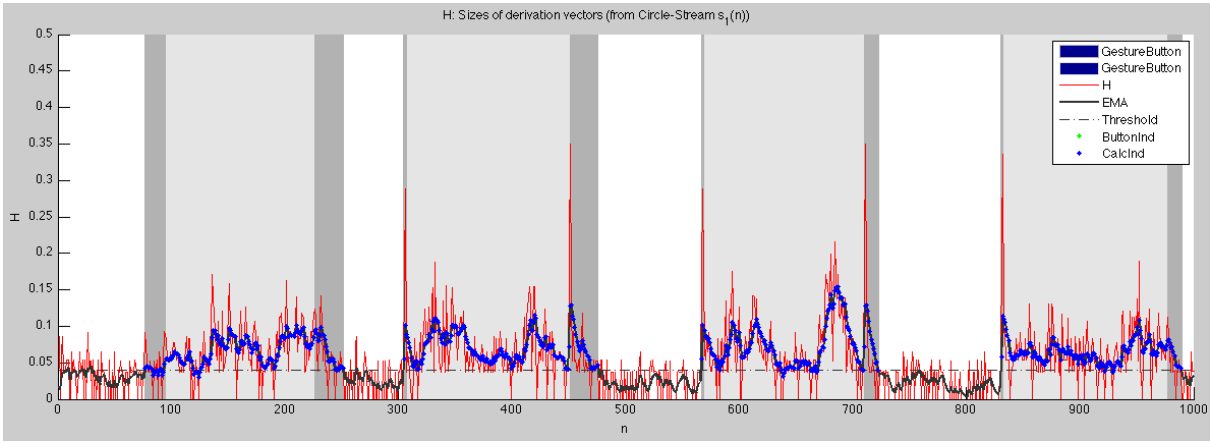

Abb. 3.27.: *Ergebnisse der Segmentierung bei Verwendung des Steigungsbetrags H und den daraus bestimmten EMA_H für die circle-Geste. Zum Vergleich sind in hellgrau die Übereinstimmungen der erkannten Gesten mit den Button-Gesten dargestellt, in dunkelgrau die Abweichungen. Zur genaueren Bestimmung der Abweichung markieren grüne Punkte den Button-Gestenindikator, blaue den Gestenindikator, der ermittelt wurde, wenn der Steigungsbetrag oberhalb eines Grenzwertes von 0.04 liegt.*

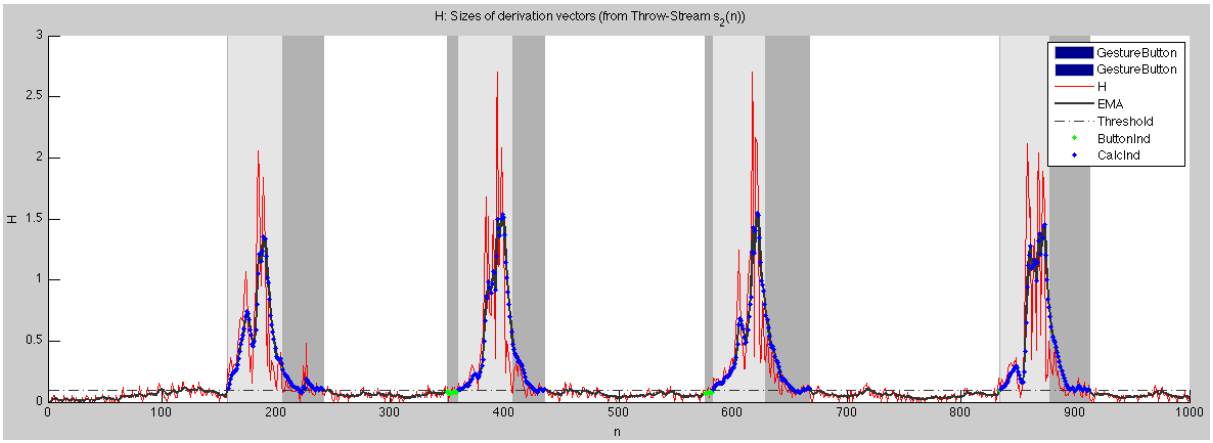

Abb. 3.28.: *Ergebnisse der Segmentierung bei Verwendung des Steigungsbetrags H und den daraus bestimmten EMA_H für die throw-Geste. Zum Vergleich sind in hellgrau die Übereinstimmungen der erkannten Gesten mit den Button-Gesten dargestellt, in dunkelgrau die Abweichungen. Zur genaueren Bestimmung der Abweichung markieren grüne Punkte den Button-Gestenindikator, blaue den Gestenindikator, der ermittelt wurde, wenn der Steigungsbetrag oberhalb eines Grenzwertes von 0.10 liegt.*

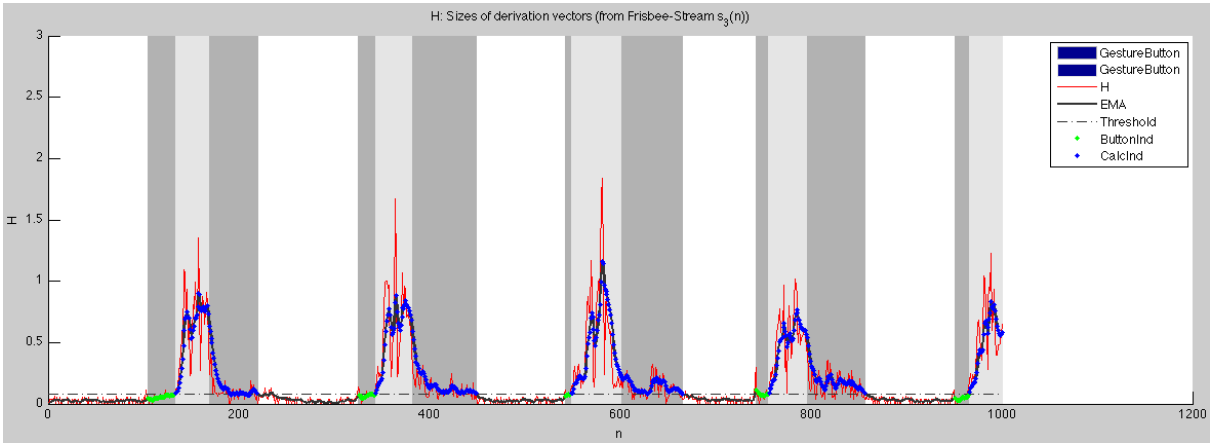

Abb. 3.29.: *Ergebnisse der Segmentierung bei Verwendung des Steigungsbetrags H und den daraus bestimmten EMA_H für die frisbee-Geste. Zum Vergleich sind in hellgrau die Übereinstimmungen der erkannten Gesten mit den Button-Gesten dargestellt, in dunkelgrau die Abweichungen. Zur genaueren Bestimmung der Abweichung markieren grüne Punkte den Button-Gestenindikator, blaue den Gestenindikator, der ermittelt wurde, wenn der Steigungsbetrag oberhalb eines Grenzwertes von 0.15 liegt.*

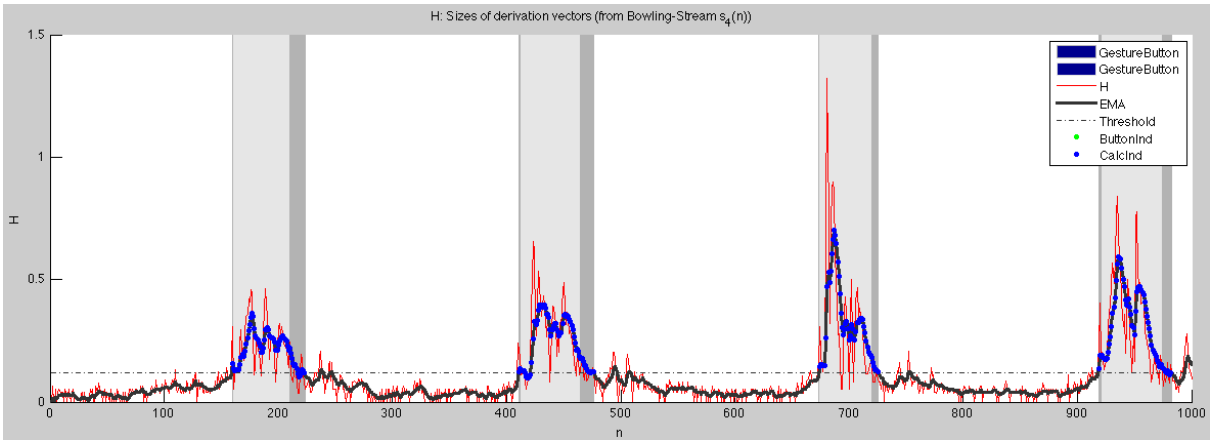

Abb. 3.30.: *Ergebnisse der Segmentierung bei Verwendung des Steigungsbetrags H und den daraus bestimmten EMA_H für die bowling-Geste. Zum Vergleich sind in hellgrau die Übereinstimmungen der erkannten Gesten mit den Button-Gesten dargestellt, in dunkelgrau die Abweichungen. Zur genaueren Bestimmung der Abweichung markieren grüne Punkte den Button-Gestenindikator, blaue den Gestenindikator, der ermittelt wurde, wenn der Steigungsbetrag oberhalb eines Grenzwertes von 0.12 liegt.*

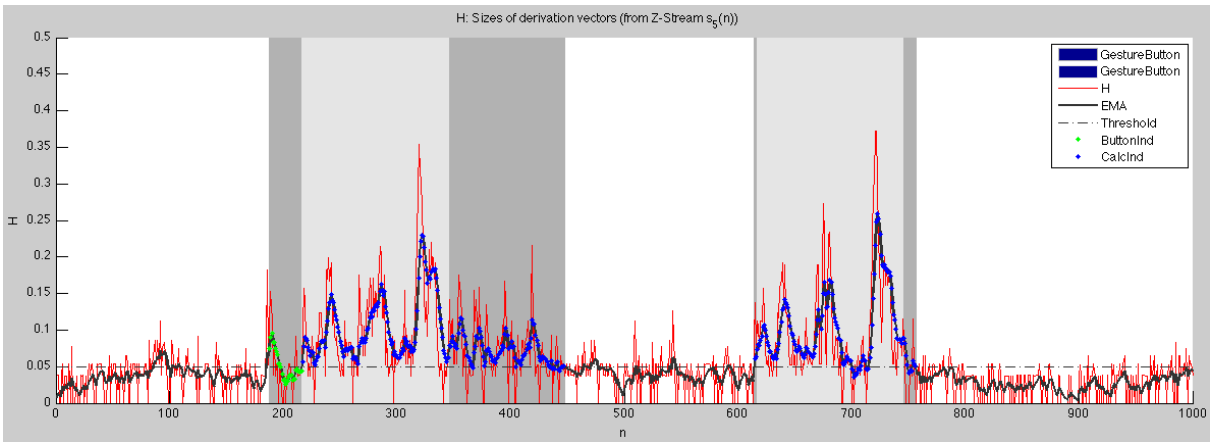

Abb. 3.31.: *Ergebnisse der Segmentierung bei Verwendung des Steigungsbetrags H und den daraus bestimmten EMA_H für die z-Geste. Zum Vergleich sind in hellgrau die Übereinstimmungen der erkannten Gesten mit den Button-Gesten dargestellt, in dunkelgrau die Abweichungen. Zur genaueren Bestimmung der Abweichung markieren grüne Punkte den Button-Gestenindikator, blaue den Gestenindikator, der ermittelt wurde, wenn der Steigungsbetrag oberhalb eines Grenzwertes von 0.05 liegt.*

3.4. Vergleich und Bewertung

Abschließend sind die Ergebnisse der Segmentierungstests in Tabelle 3.10 nochmals zusammengefasst. Insgesamt konnten mit der regelbasierten Segmentierung (Hysterese) die besten Übereinstimmungen mit dem Button-Gestenindikator erzielt werden.

Tab. 3.10.: *Gegenüberstellung der Abweichungen Var in den untersuchten Segmentierungsverfahren*

gesture	$I_{RB}calib$	$I_{RB}nc$	$I_{SFA}r$	$I_{SFA}acc$	$I_{SFA}2x$	I_{EMA}
circle	**6.21%**	6.72%	6.73%	6.47%	6.97%	7.25%
throw	8.08%	9.00%	7.36%	**6.10%**	6.82%	15.42%
frisbee	**8.84%**	10.93%	10.40%	11.24%	11.42%	16.79%
bowling	**2.47%**	4.11%	4.15%	5.28%	4.74%	8.14%
z	3.55%	**3.21%**	3.26%	3.80%	3.19%	7.25%

In den folgenden Abbildungen sind die drei vorgestellten Segmentierungsverfahren für die *circle*-Geste nochmals im direkten Vergleich gegenübergestellt:

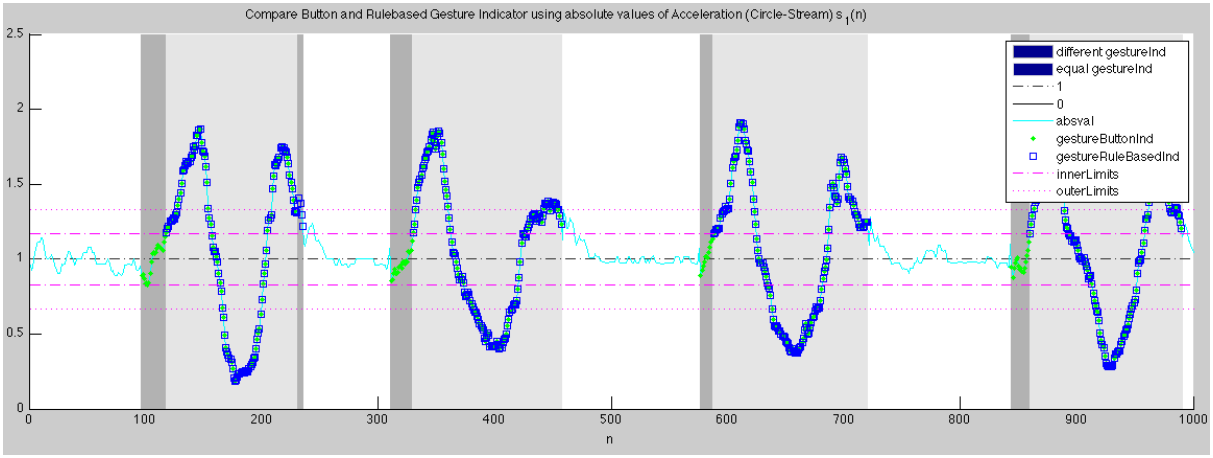

Abb. 3.32.: *Vergleich Segmentierung: Regelbasierte Segmentierung von r_{circle} + Button-Gestenindikator*

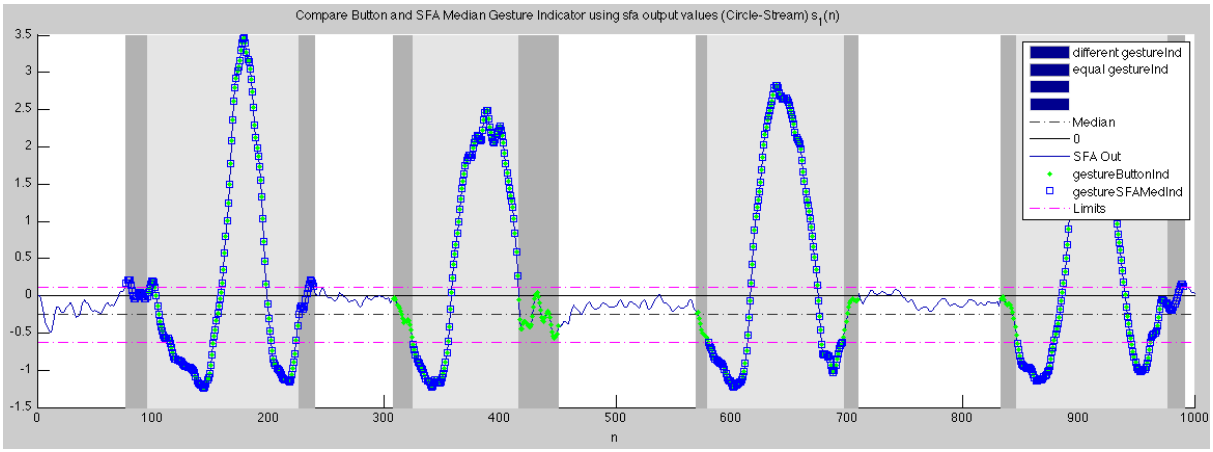

Abb. 3.33.: *Vergleich Segmentierung: Segmentierung mit einfacher SFA mit $acc_{circle}, m = 15$ + Button-Gestenindikator*

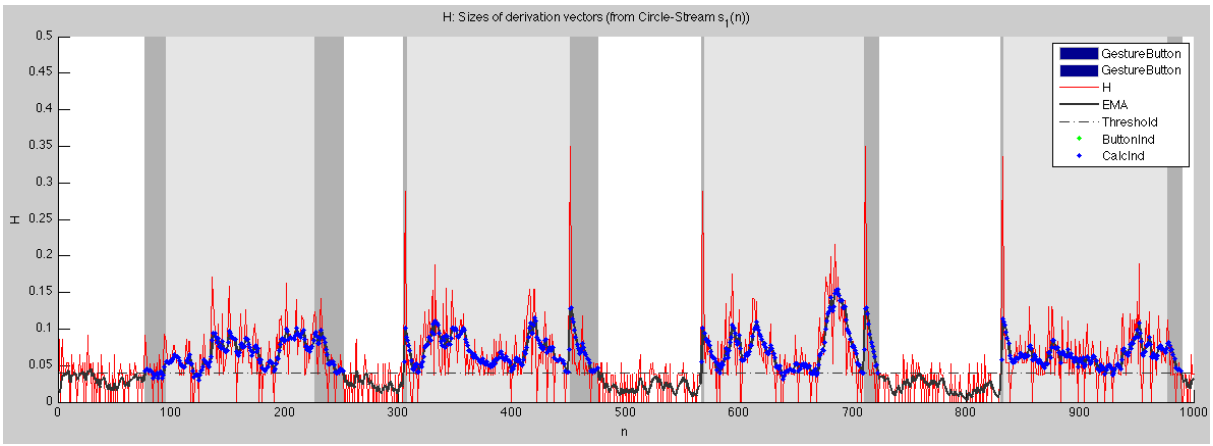

Abb. 3.34.: *Vergleich Segmentierung: Segmentierung über EMA_{circle} + Button-Gestenindikator*

- Bei allen untersuchten Verfahren mussten abhängig von der Geste die optimalen Parameter bestimmt werden. Diese lagen teilweise in sehr unterschiedlichen Bereichen. Dadurch konnten keine einheitlichen Parameter übergreifend für alle Gesten bestimmt werden, mit denen akzeptable Ergebnisse erzielt wurden.

- Bei der regelbasierten Segmentierung und der Segmentierung mit der SFA werden bei den als optimal gefundenen Parametern eher kürzere Gesten erkannt als sie durch den Button markiert wurden. Beim Ansatz über den Steigungsbetrag sind die gefundenen Gesten i.d.R. länger als die Button-Gesten.

- Durch zusätzliche Tests lassen sich für die einzelnen Ansätze möglicherweise einheitliche Einstellungen finden. Dazu sollten aber auch Aufzeichnungen von weiteren Personen verwendet werden.

- Zukünftig könnte auch anhand von Metaregeln versucht werden die Parameter auf möglicht geringe Abweichungen zum Button-Gestenindikator zu tunen, anstatt einzelne Parameter experimentell zu ermitteln.

- Allerdings besteht bei den für jede Geste optimierten Parametern das Problem, dass im praktischen Einsatz vorher bekannt sein müsste welche Geste verwendet wird. Um dieses Problem zu beheben, müssten für alle Gesten übergreifende Einstellungen verwendet werden.

- Bei dem Ansatz über die Beschleunigungsänderung hätten vielleicht bei weiterem Tuning der Einstellungen noch bessere Ergebnisse erzielt werden können (aus Zeitgründen ist dies bisher nicht geschehen).

- Als Referenz wurden die Button-Gesten betrachtet. Diese müssen auch nicht immer exakt sein, weil dieser Gestenindikator zum Einen die subjektiven Gestengrenzen der jeweiligen Person wiederspiegelt und zum Anderen aus rein motorischen Gründen die Grenzen nicht 100% exakt markiert werden. Aus diesem Grund lässt sich mit der Referenz über die Button-Gesten nur abschätzen, ob ein Segmentierungsverfahren besser geeignet ist als ein anderes.

- Eine Segmentierung mit dem SFA-Output ist mit entsprechenden Parametereinstellungen grundsätzlich möglich. Es macht dabei wenig Unterschied, ob dabei der Beschleunigungsbetrag oder die Beschleunigungsrohdaten verwendet werden, da aufgrund der Expansionseigenschaft der SFA dieser Beschleunigungsbetrag indirekt im expandierten Eingangsvektor enthalten ist. Die doppelte Anwendung der SFA "glättet" die Ergebnisse etwas, bringt aber für die Segmentierung keine neuen Erkenntnisse.

- Da für alle Verfahren ein Parametertuning notwendig ist, muss allerdings die Frage gestellt werden, ob zur Segmentierung tatsächlich die SFA eingesetzt werden soll, oder nicht lieber auf ein simpleres Verfahren, das weniger Laufzeit benötigt, zurückgegriffen werden sollte. Auf Basis der bisher untersuchten Segmentierungsmöglichkeiten mit der SFA sollte für einen gemeinsamen Einsatz der Segmentierung mit anschließender Klassifizierung zur Laufzeit auf jeden Fall ein schnelleres Verfahren, wie z.B. Prekopcsák [Pre08] es vorschlägt, oder eine regelbasierte Trennung über den Beschleunigungsbetrag verwendet werden.

- Bisher wurde noch nicht der "richtig" Weg zur Segmentierung mit der SFA gefunden. Mögliche Wege durch eine andere Vorverarbeitung mit der die unterschiedlichen Gestensegmente ähnlicher werden, müssten in weitergehenden Untersuchungen betrachtet werden.

- Wie gut sich die einzelnen Verfahren tatsächlich im Gesamtprozess zur Gestenerkennung eignen, müsste in weiteren Untersuchungen getestet werden, in dem man die gefundenen Gesten-Segmente der einzelnen Verfahren als Testdaten für die Klassifizierung verwendet. Dabei bräuchten nicht zwangsläufig nur die Abweichungen zu einem Referenzgestenindikator betrachtet werden, sondern es würde der Informationsverarbeitungsprozess insgesamt betrachtet. Möglicherweise kristallisiert sich dabei heraus, welches Verfahren tatsächlich am besten zur Segmentierung geeignet ist.

4. Fazit

In dieser Arbeit ist eine mögliche Verwendung der Slow Feature Analysis (SFA) zur Gestenerkennung untersucht worden. Zum Einen wurde eine personen(un)abhängige Klassifizierung mit der SFA betrachtet, zum Anderen die Segmentierung, bei der die Gestensegmente aus einem Zeitsignal mit abwechselnder Geste und Nicht-Geste herausgefiltert wurden. Zur Bewertung der Gestenerkennung mit SFA wurden vergleichend andere Verfahren aus den beiden Bereichen der Gestensegmentierung und -klassifizierung herangezogen.

Die Untersuchungen dieser Arbeit haben gezeigt, dass sich die SFA, obwohl sie aus einer anderen Domäne stammt, zur Klassifizierung von Gesten eignet. Sie ist vergleichbar mit anderen Verfahren und hat sogar in direkten Vergleichstests mit einem gängigen Verfahren aus dem Bereich der Mustererkennung, dem Hidden Markov Modell (HMM), bessere Ergebnisse erzielt. Es war möglich, mit der SFA zusätzlich zur personenabhängigen Klassifizierung, wie es mit dem HMM möglich ist, auch personenübergreifend eine gute Klassifikation zu erreichen. Gezeigt wurde außerdem, dass eine Klassifizierung mit SFA unter Verwendung von Parametric Bootstrap auch mit wenigen Trainingsdaten pro Geste funktioniert.

Generell scheint die SFA für den Bereich der Mustererkennung vielversprechend, da sie nicht wie andere Verfahren statistisch die Ähnlichkeitswahrscheinlichkeit von Mustern bestimmt, sondern dadurch, dass sie die am langsamsten variierenden Merkmale aus dem Signal selbst findet und diese Merkmale innerhalb von Mustern gleicher Klassen ähnlich sind. Dadurch lässt sie sich in vielen Bereichen einsetzen und ist nicht auf einen speziellen Problembereich beschränkt. Außerdem sind für den Einsatz der SFA nahezu keine Parametereinstellungen notwendig, da sie die Merkmale aus den gegebenen Eingangssignalen selbst extrahiert und ordnet. Sie kann damit sowohl in der Gestenerkennung als auch in anderen Bereichen der Mustererkennung Verwendung finden.

Eine automatische Segmentierung mit Einsatz der SFA ist zwar möglich, allerdings sollte für den interaktiven Einsatz ein schnelleres Verfahren bevorzugt werden. Alle untersuchten Verfahren zur Segmentierung, inklusive der SFA, benötigen ein Feintuning der jeweiligen Parametereinstellungen, das auf die verwendeten Gesten entsprechend abgestimmt ist. Selbst wenn sich beim Parametertuning optimale Werte finden ließen, so sind diese doch immer noch von dem jeweiligen verwendeten Gestenset abhängig und variieren möglicherweise auch von Person zu Person. Dies sind auch die Kritikpunkte an dieser Arbeit. Es wurde nur ein relativ kleines Set von Gesten, und für die Segmentierung sogar nur die Gesten einer Person, verwendet. Diese wurden zudem noch "in vitro" erhoben, wodurch sie möglicherweise nicht exakt der Verwendung in realen Umgebungen entsprechen. Für weiterführende Untersuchungen wären zusätzlich andere, vielleicht auch nicht leicht separierbare Gesten notwendig.

In zukünftigen Arbeiten sollte außerdem der Gesamtprozess der Gestenerkennung, beginnend mit der Segmentierung und der anschließenden Klassifizierung mit der SFA, genauer untersucht werden. Wird die SFA dabei mit einem festen Gestenset vortrainiert, lässt sich voraussichtlich eine interaktive Gestenerkennung mit der SFA umsetzen.

Heutzutage werden in vielen gebräuchlichen Geräten Beschleunigungssensoren verwendet, wie z.B. in Mobiltelefonen. Hiermit erschließt sich ein breites Einsatzgebiet auf dem aktuellen Markt, in dem eine Gestenerkennung notwendig ist. Eine gute Gestenerkennung könnte eine zusätzliche Kommunikationsmodalität zwischen Mensch und Maschine bieten und z.B. zur Steuerung von Anwendungen oder als Übersetzer von Gesten in andere Kodalitäten eingesetzt werden, z.B. in natürliche Sprache.

A. Anhang

A.1. Entstandene Artefakte

Im Rahmen der Masterarbeit sind folgende Dokumente in Form von Quellcode, Skripten, etc. entstanden oder erweitert worden:

- ein javabasiertes WiiTool wie in 2.3.1 beschrieben, das im Rahmen der Masterthesis durch den zweiten Aufzeichnungsmodus, eine nochmals überarbeitete Datenstruktur, ein zusätzliches Wiimote(Device)-Objekt mit deaktivierter Vorfilterung, eine (Re-)Kalibrierung der Daten und das Testmodul erweitert wurde
- diverse Matlab-Skripte zur Klassifizierung:
  - zum Einlesen der erhobenen Daten von 10 Probanden und die Implementierung einer Vorverarbeiten der Daten für die Weiterverwendung in Matlab
  - zum Testen der Klassifizierung mit SFA mit personenabhängigen Tests, Tests über alle Daten, Tests einer ungesehenen Person, Test mit wenigen Trainings inklusive der grafischen Ausgaben der Ergebnisse
  - zum Test der einzelnen PB-Verfahren und deren grafische Gegenüberstellung
- diverse Matlab-Skripte zur Segmentierung:
  - zum Einlesen Geste-/Nicht-Geste-Zeitreihen, die hierfür im Rahmen der Masterarbeit erhoben wurden und eine Vorverarbeiten der Daten in Form einer Rekalibrierung wie sie auch im WiiTool verwendet wird
  - zum Testen der Segmentierung durch das Regelbasierte Verfahren, inklusive unterschiedlicher grafischer Darstellungen
  - zum Testen der Segmentierung durch SFA in unterschiedlichen Varianten, inklusive unterschiedlicher grafischer Darstellungen
  - zum Testen der Segmentierung über die durchschnittliche Bewegungsänderung, inklusive unterschiedlicher grafischer Darstellungen

A.2. Ergebnisse des Klassifikationstest III mit HMM

Im Folgenden ist eine detaillierte Auflistung der einzelnen HMM Testläufe zur personenabhängigen Klassifizierung mit HMM dargestellt.

Tab. A.1.: *Ergebnisse der einzelnen Testläufe zur personenabhängig Klassifizierung mit HMM*

run \person	121	122	123	124	125
Run1	31,88% ± 12,43%	45,95% ± 24,47%	14,55% ± 29,78%	17,86% ± 12,96%	30,77% ± 18,47%
Run2	30,43% ± 15,67%	43,24% ± 15,44%	14,55% ± 20,85%	17,86% ± 16,71%	26,15% ± 21,32%
Run3	33,33% ± 17,37%	47,30% ± 20,54%	10,91% ± 13,36%	17,86% ± 19,55%	26,15% ± 14,48%
Run4	27,54% ± 13,60%	45,95% ± 13,09%	12,73% ± 13,06%	21,43% ± 16,39%	26,15% ± 22,30%
Run5	28,99% ± 24,75%	48,65% ± 17,41%	23,64% ± 10,85%	17,86% ± 18,30%	27,69% ± 15,51%
Run6	36,23% ± 17,44%	43,24% ± 16,92%	9,09% ± 11,21%	19,64% ± 16,87%	24,62% ± 27,23%
Run7	36,23% ± 19,81%	43,24% ± 21,20%	12,73% ± 10,72%	17,86% ± 12,96%	27,69% ± 14,14%
Run8	30,43% ± 19,18%	44,59% ± 16,22%	12,73% ± 13,06%	17,86% ± 14,95%	24,62% ± 20,79%
Run9	30,43% ± 21,12%	39,19% ± 15,39%	10,91% ± 11,09%	21,43% ± 17,10%	23,08% ± 14,15%
Run10	36,23% ± 23,43%	40,54% ± 20,07%	14,55% ± 16,37%	19,64% ± 13,17%	26,15% ± 18,28%
mean	**32,17%** ± 3,03%	**44,19%** ± 2,77%	**13,64%** ± 3,75%	**18,93%** ± 1,43%	**26,31%** ± 2,00%
run \person	126	127	128	129	130
Run1	24,53% ± 17,39%	34,33% ± 20,31%	21,43% ± 13,17%	17,86% ± 12,96%	35,76% ± 6,39%
Run2	26,42% ± 20,01%	37,31% ± 16,14%	22,86% ± 15,91%	25,00% ± 23,86%	27,81% ± 13,35%
Run3	18,87% ± 12,70%	37,31% ± 13,38%	21,43% ± 11,52%	21,43% ± 25,02%	31,13% ± 18,98%
Run4	18,87% ± 9,23%	35,82% ± 18,53%	21,43% ± 9,58%	19,64% ± 20,39%	33,11% ± 10,84%
Run5	18,87% ± 15,66%	31,34% ± 16,76%	18,57% ± 11,16%	23,21% ± 15,47%	31,79% ± 16,61%
Run6	26,42% ± 16,55%	35,82% ± 16,17%	18,57% ± 15,71%	21,43% ± 16,39%	34,44% ± 16,04%
Run7	20,75% ± 17,95%	34,33% ± 23,58%	21,43% ± 14,64%	23,21% ± 17,17%	33,77% ± 12,24%
Run8	24,53% ± 20,08%	35,82% ± 13,73%	20,00% ± 9,48%	19,64% ± 15,14%	31,79% ± 9,39%
Run9	22,64% ± 30,91%	37,31% ± 17,52%	22,86% ± 13,09%	21,43% ± 21,43%	33,77% ± 14,04%
Run10	22,64% ± 24,21%	37,31% ± 13,38%	18,57% ± 14,36%	21,43% ± 19,49%	32,45% ± 11,00%
mean	**22,45%** ± 2,86%	**35,67%** ± 1,82%	**20,71%** ± 1,60%	**21,43%** ± 1,96%	**32,58%** ± 2,07%

A.3. Regelbasierte Segmentierung (Hysterese)

Im Folgenden sind der Ergebnisse der regelbasierte Segmentierung über Hysterese-Funktion mit rekalibrierten Daten dargestellt. Eingestellt wurden die als optimal ermittelten Parameter (siehe 3.4)und hier im Vergleich mit dem Button-Gestenindikator geplottet. Beschreibung siehe 3.3.1.

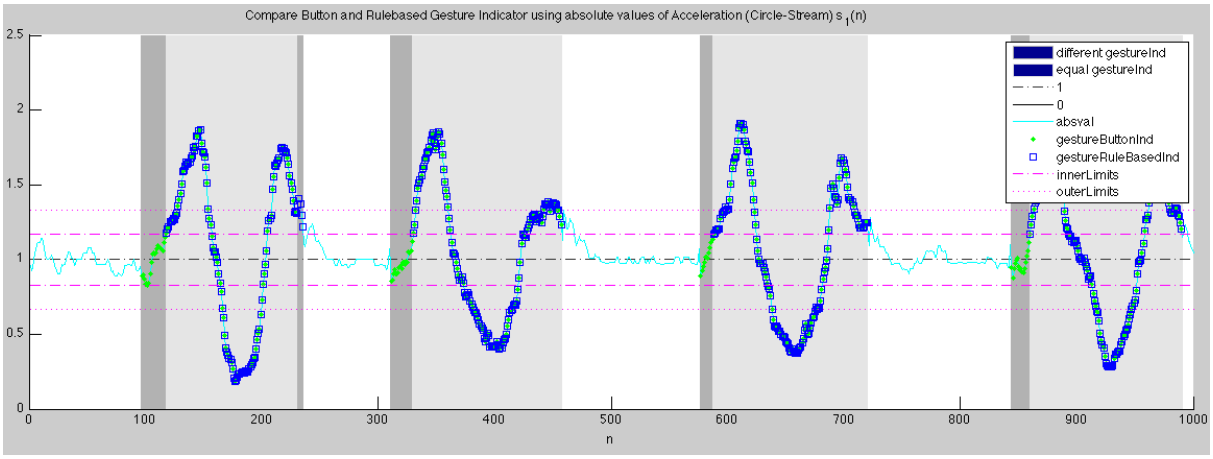

Abb. A.1.: *Regelbasierte Segmentierung von r (circle-Geste) + Button-Gestenindikator*

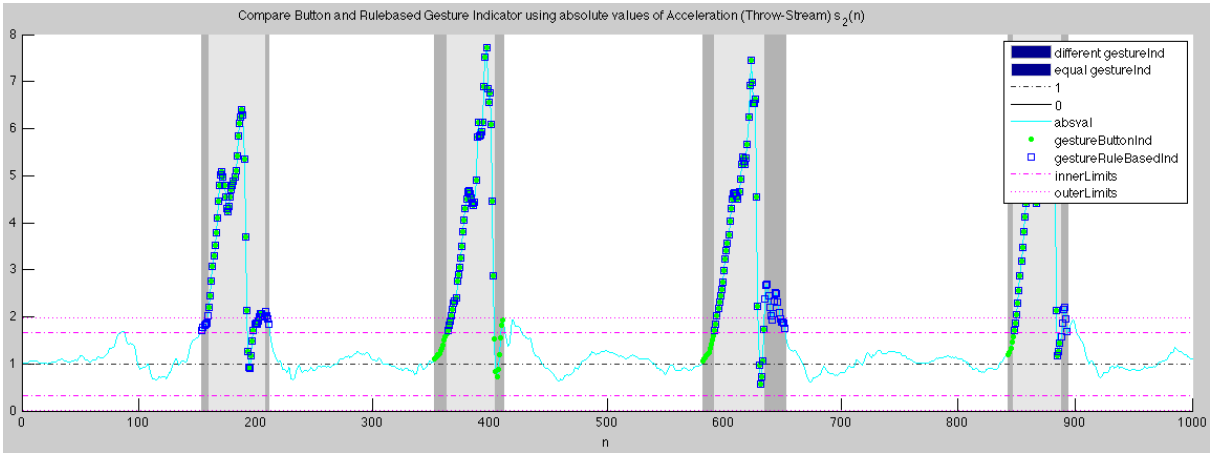

Abb. A.2.: *Regelbasierte Segmentierung von r (throw-Geste) + Button-Gestenindikator*

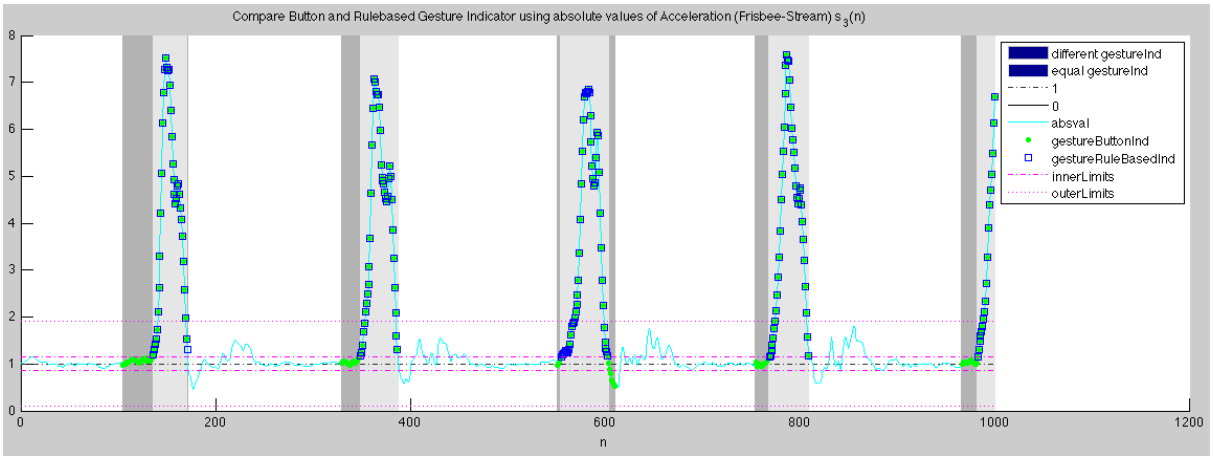

Abb. A.3.: *Regelbasierte Segmentierung von r (frisbee-Geste) + Button-Gestenindikator*

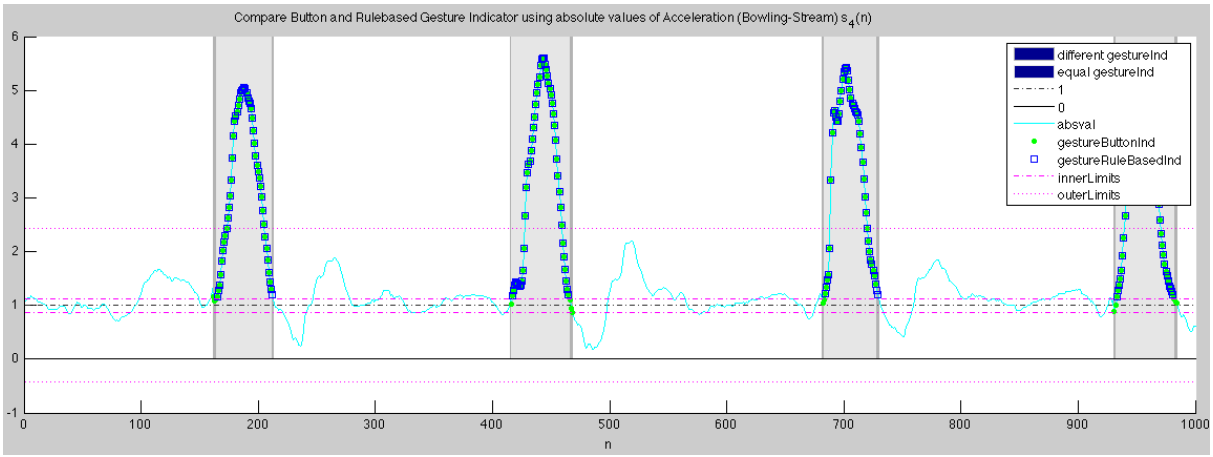

Abb. A.4.: *Regelbasierte Segmentierung von r (bowling-Geste) + Button-Gestenindikator*

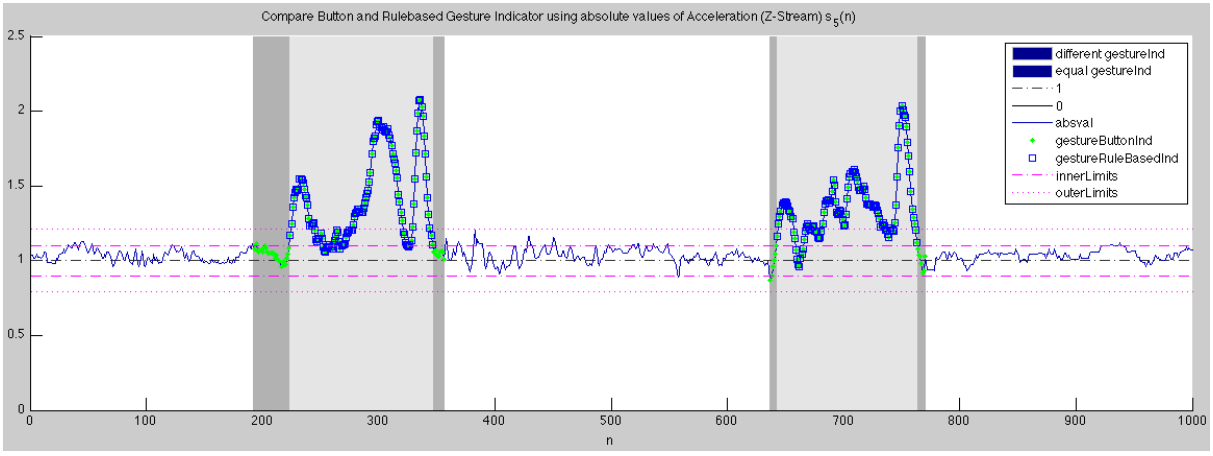

Abb. A.5.: *Regelbasierte Segmentierung von r (z-Geste) + Button-Gestenindikator*

A.4. Ergebnisse zur Voruntersuchung: Segmentierung einfache SFA mit Beschleunigungsbetrag

Im Folgenden sind die Ergebnisse der Voruntersuchung zur Segmentierung bei Verwendung einer einfach durchgeführten SFA mit dem Beschleunigungsbetrag als Eingangssignal grafisch für alle fünf Gesten dargestellt. Nähere Beschreibung siehe Abschnitt "einfache SFA mit Beschleunigungsbetrag" in 3.3.2.

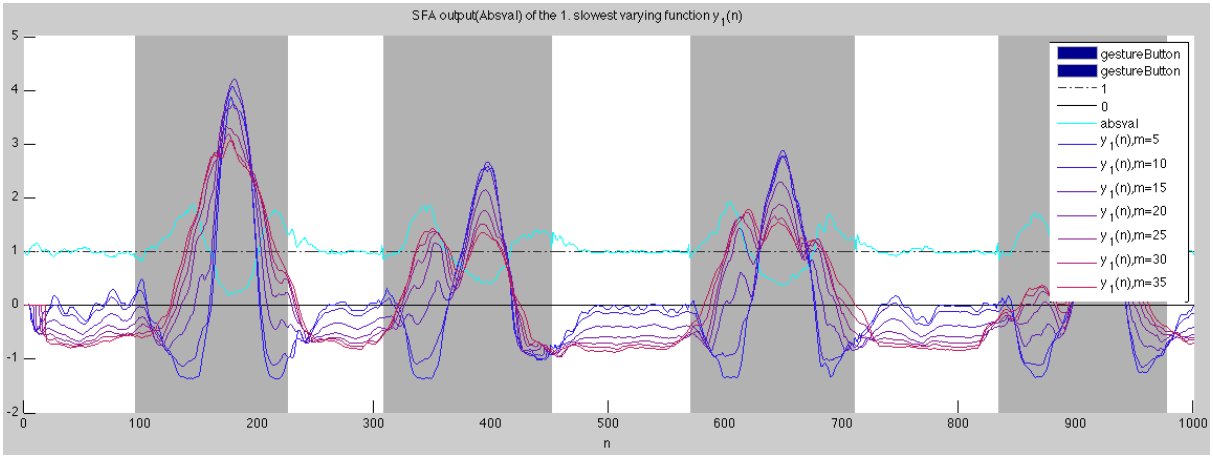

Abb. A.6.: *Segmentierung mit SFA über Beschleunigungsbetrag: Beschleunigungsbetrag r (circle-Geste) als Input und SFA Output (embedding-dim:5-35)*

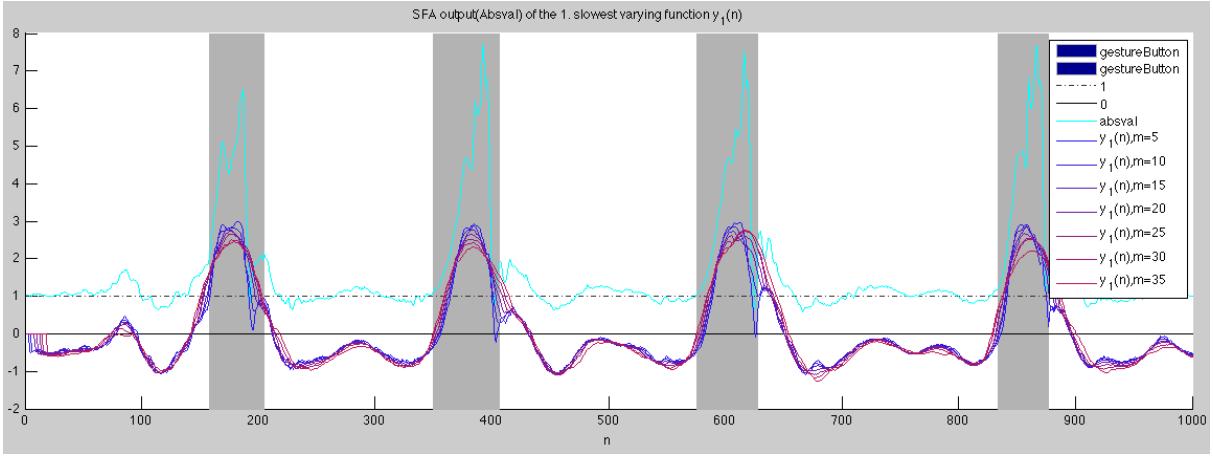

Abb. A.7.: *Segmentierung mit SFA über Beschleunigungsbetrag: Beschleunigungsbetrag r (throw-Geste) als Input und SFA Output (embedding-dim:5-35)*

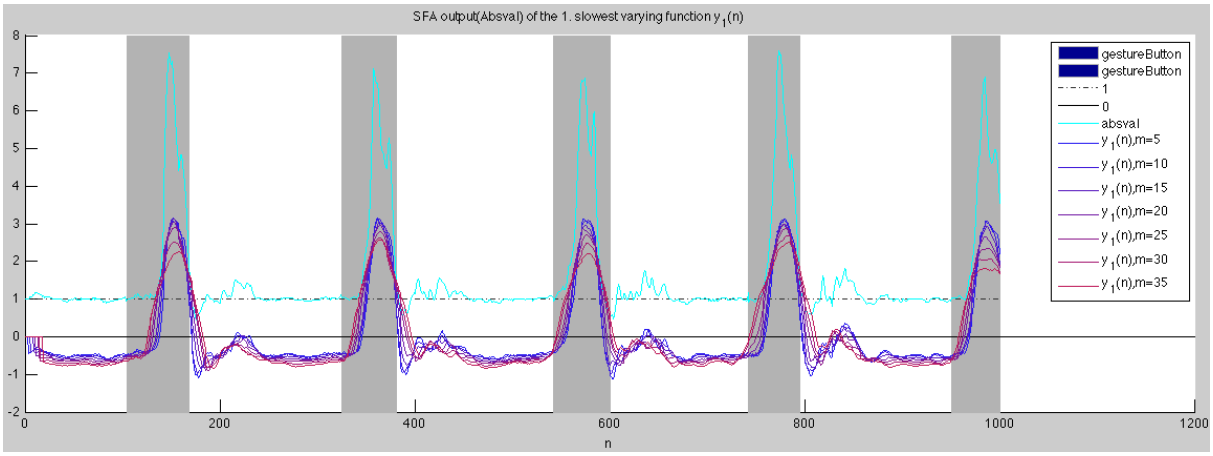

Abb. A.8.: *Segmentierung mit SFA über Beschleunigungsbetrag: Beschleunigungsbetrag r (frisbee-Geste) als Input und SFA Output (embedding-dim:5-35)*

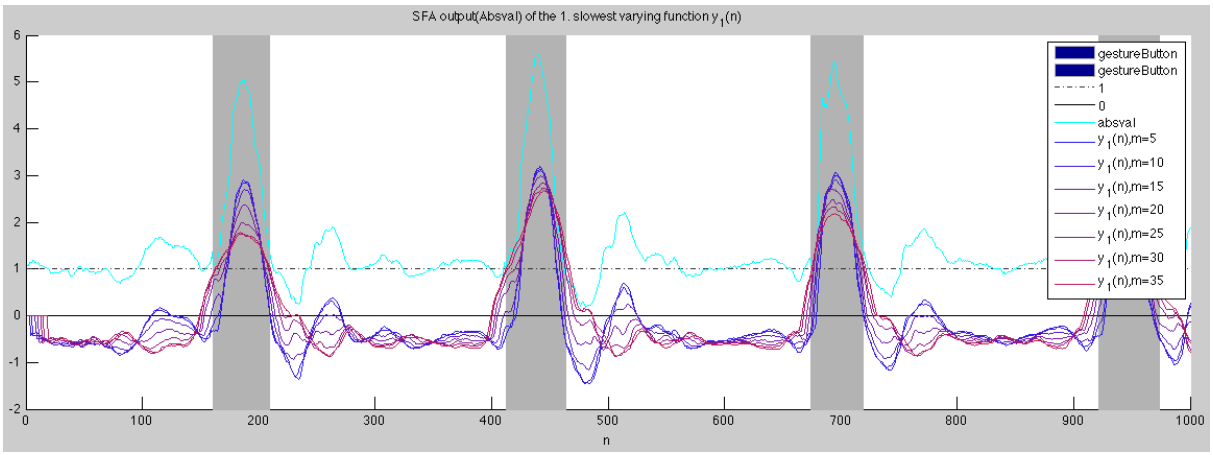

Abb. A.9.: *Segmentierung mit SFA über Beschleunigungsbetrag: Beschleunigungsbetrag r (bowling-Geste) als Input und SFA Output (embedding-dim:5-35)*

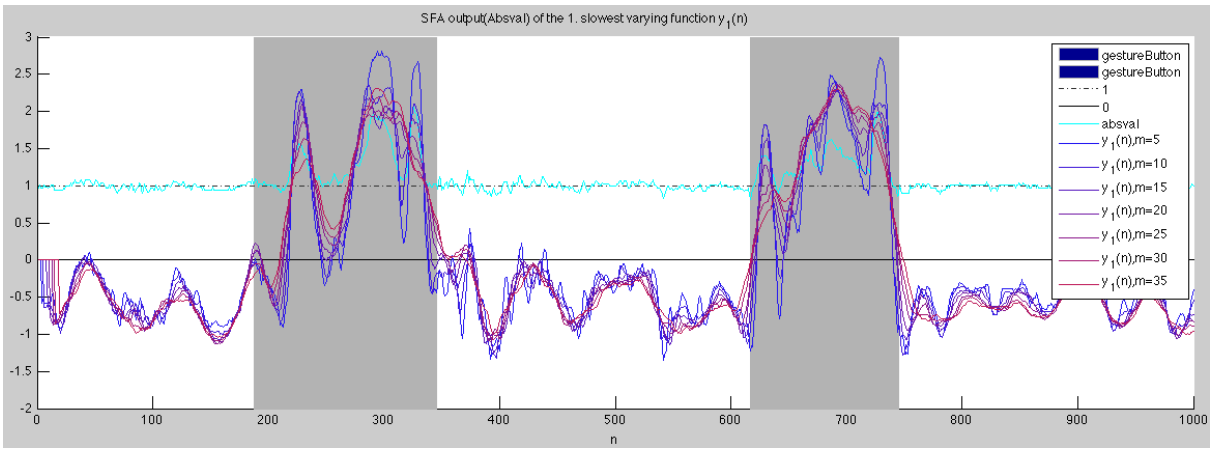

Abb. A.10.: *Segmentierung mit SFA über Beschleunigungsbetrag: Beschleunigungsbetrag r (z-Geste) als Input und SFA Output (embedding-dim:5-35)*

A.5. Ergebnisse zur Voruntersuchung: Segmentierung einfache SFA mit Beschleunigungsrohdaten

Darstellung des SFA-Segmentierung mit Inputs und Outputs

In diesem Abschnitt sind die Ergebnisse der Voruntersuchung zur Segmentierung bei Verwendung einer einfache SFA mit dem Beschleunigungsrohdaten acc_x, acc_y und acc_z als Eingangssignal grafisch für alle fünf Gesten dargestellt. Nähere Beschreibung siehe Abschnitt "einfach SFA mit Beschleunigungsrohdaten" in 3.3.2.

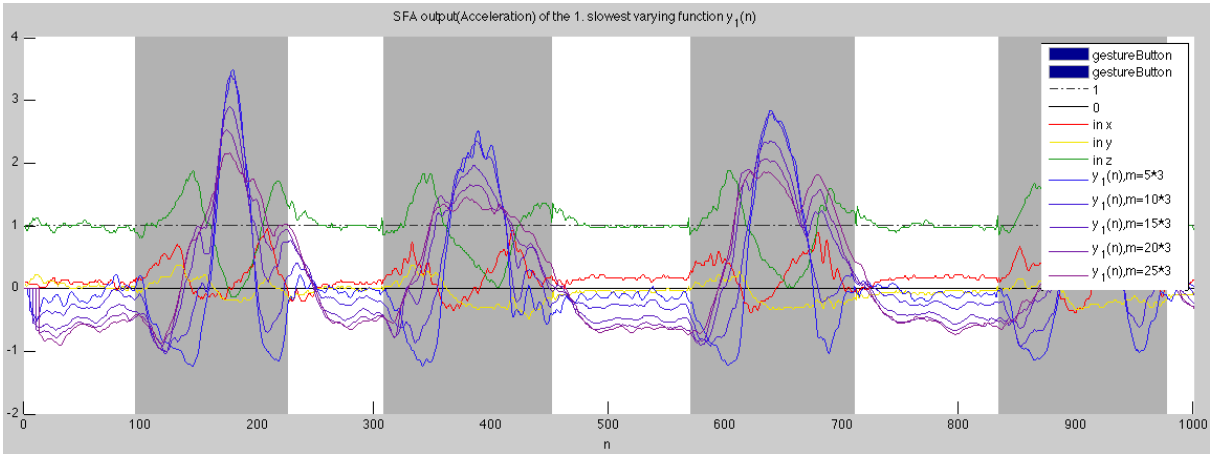

Abb. A.11.: *Segmentierung mit SFA (Vorüberlegung): Beschleunigungsrohdaten acc (circle-Geste) als Input und SFA Output (embedding-dim:15-75)*

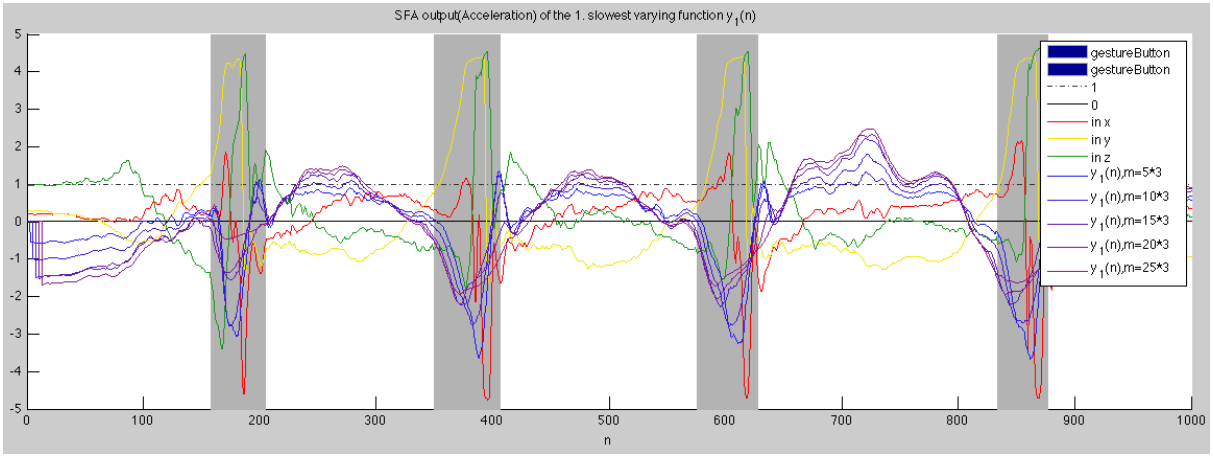

Abb. A.12.: *Segmentierung mit SFA (Vorüberlegung): Beschleunigungsrohdaten acc (throw-Geste) als Input und SFA Output (embedding-dim:15-75)*

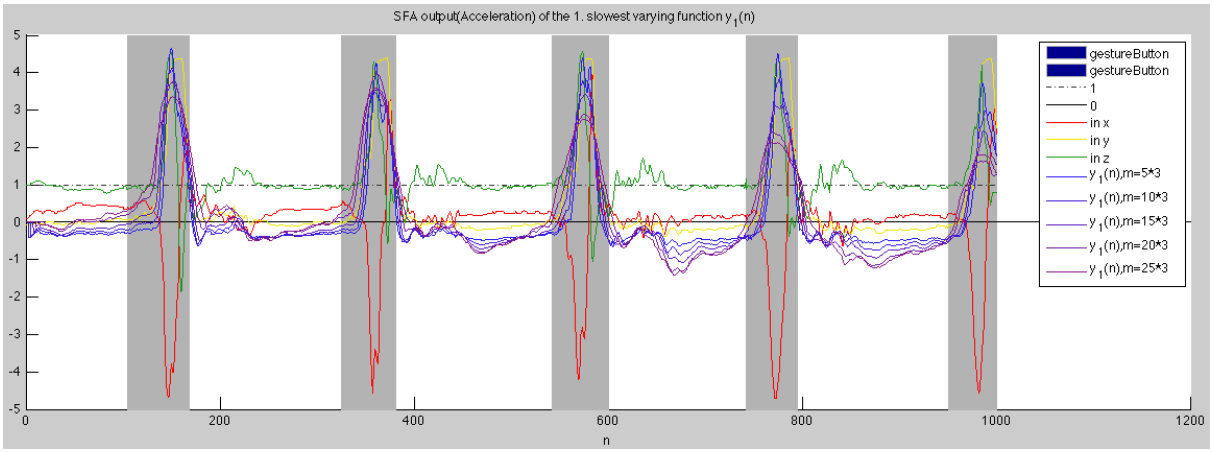

Abb. A.13.: *Segmentierung mit SFA (Vorüberlegung): Beschleunigungsrohdaten acc (frisbee-Geste) als Input und SFA Output (embedding-dim:15-75)*

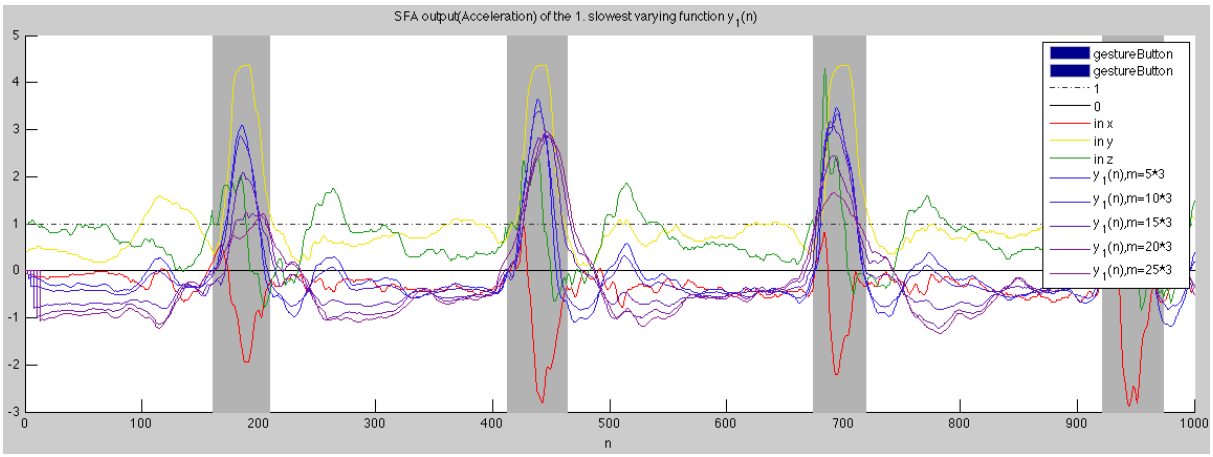

Abb. A.14.: *Segmentierung mit SFA (Vorüberlegung): Beschleunigungsrohdaten acc (bowling-Geste) als Input und SFA Output (embedding-dim:15-75)*

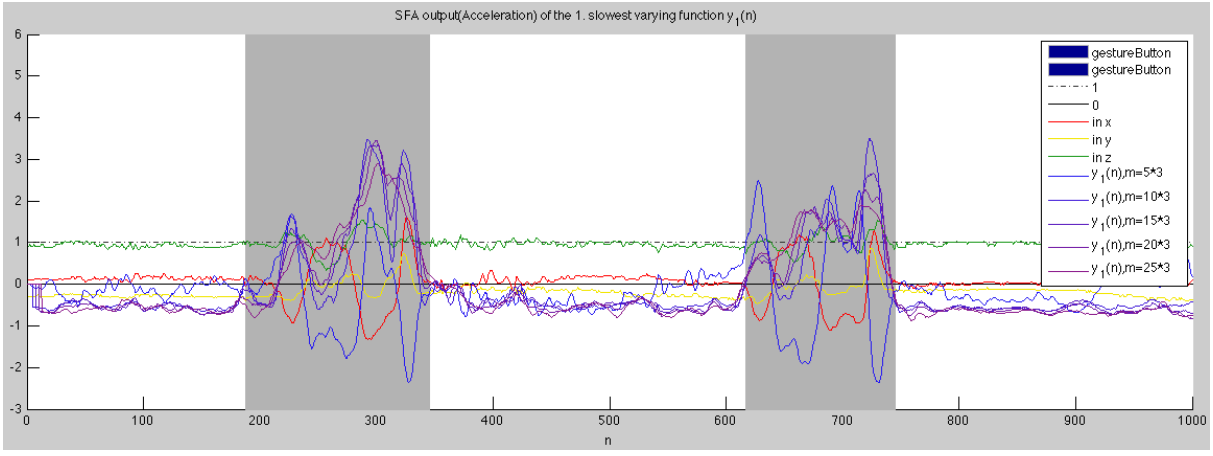

Abb. A.15.: *Segmentierung mit SFA (Vorüberlegung): Beschleunigungsrohdaten acc (z) als Input und SFA Output (embedding-dim:15-75)*

Darstellung des SFA Outputs mit Median

In den folgenden Abbildungen ist für alle fünf Gesten jeweils das des langsamsten von der SFA gefundene Signal für die *embedding*-Dimensionen 15-75 dargestellt. Eingangssignal waren die Beschleunigungswerte acc_x, acc_y und acc_z. Zusätzlich wurde der Median jedes SFA-Ausgangssignal, der zur Segmentierung der SFA verwendet wird, geplottet.

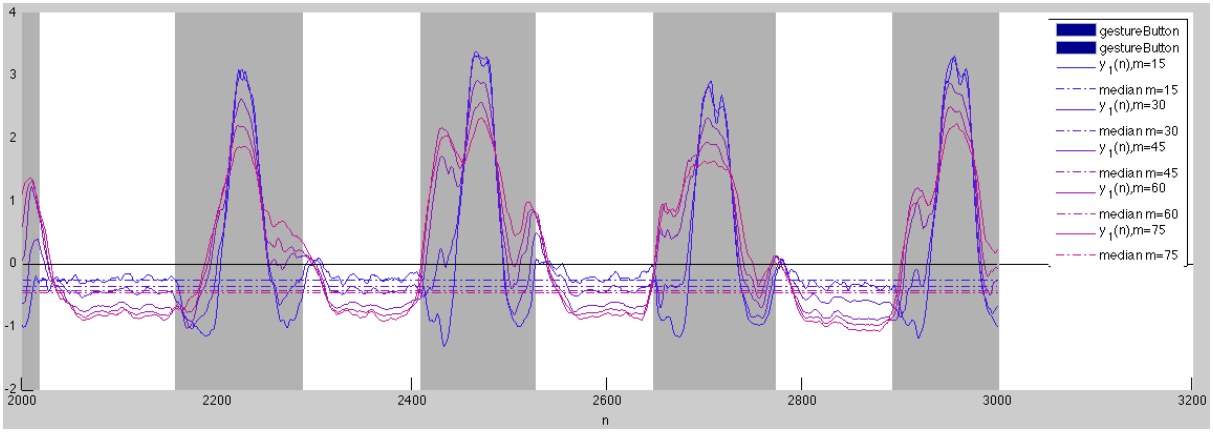

Abb. A.16.: *Segmentierung mit einfacher SFA über Beschleunigungsrohdaten* acc_{circle_c}*: 1.slowest (embedding-dim:15-75) +* $Median_c$ *(mirrored)*

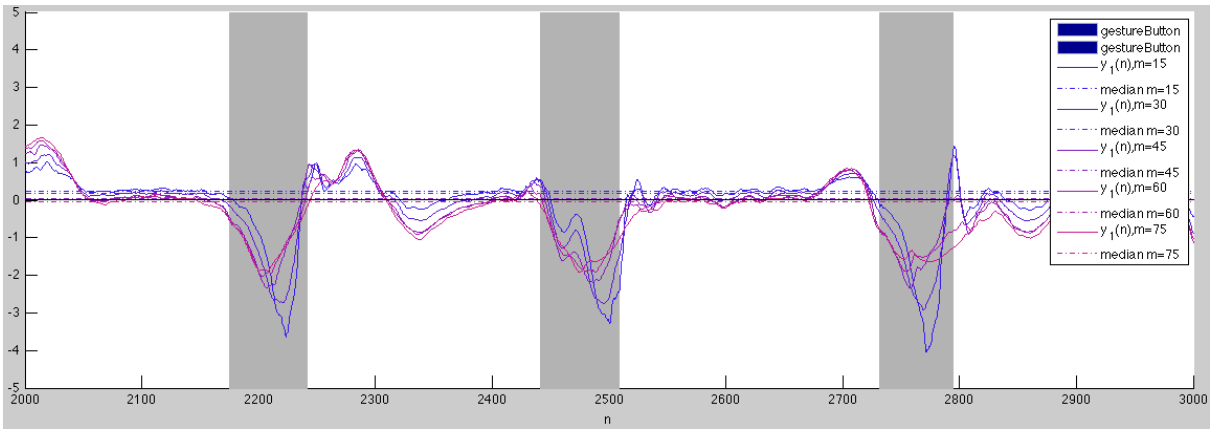

Abb. A.17.: *Segmentierung mit einfacher SFA über Beschleunigungsrohdaten* acc_{throw_c}*: 1.slowest (embedding-dim:15-75) +* $Median_c$ *(mirrored)*

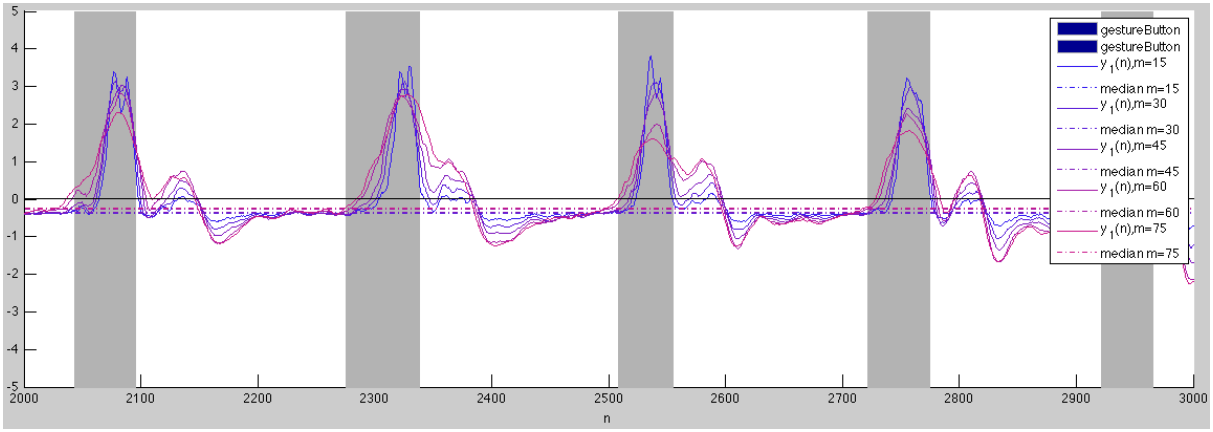

Abb. A.18.: *Segmentierung mit einfacher SFA über Beschleunigungsrohdaten $acc_{frisbee_c}$: 1.slowest (embedding-dim:15-75) + $Median_c$ (mirrored)*

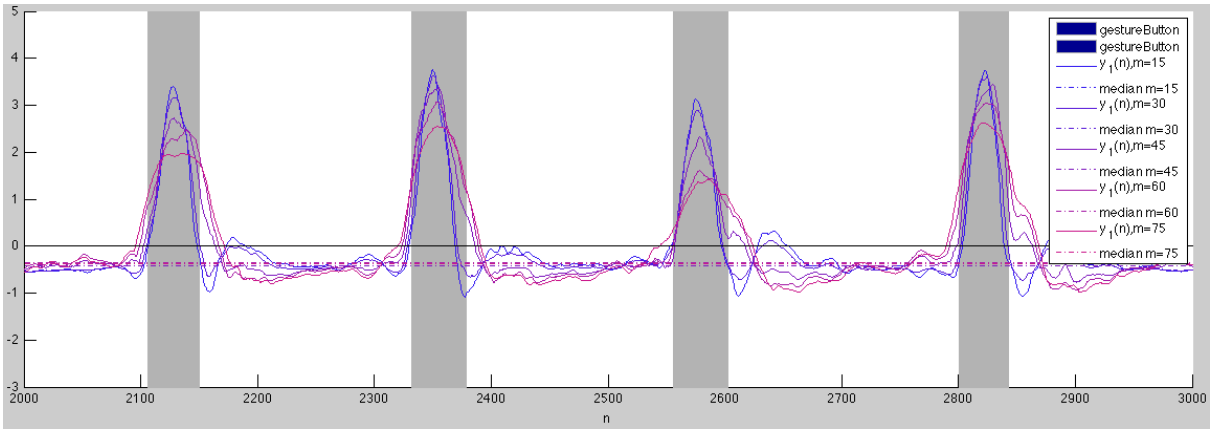

Abb. A.19.: *Segmentierung mit einfacher SFA über Beschleunigungsrohdaten $acc_{bowling_c}$: 1.slowest (embedding-dim:15-75) + $Median_c$ (mirrored)*

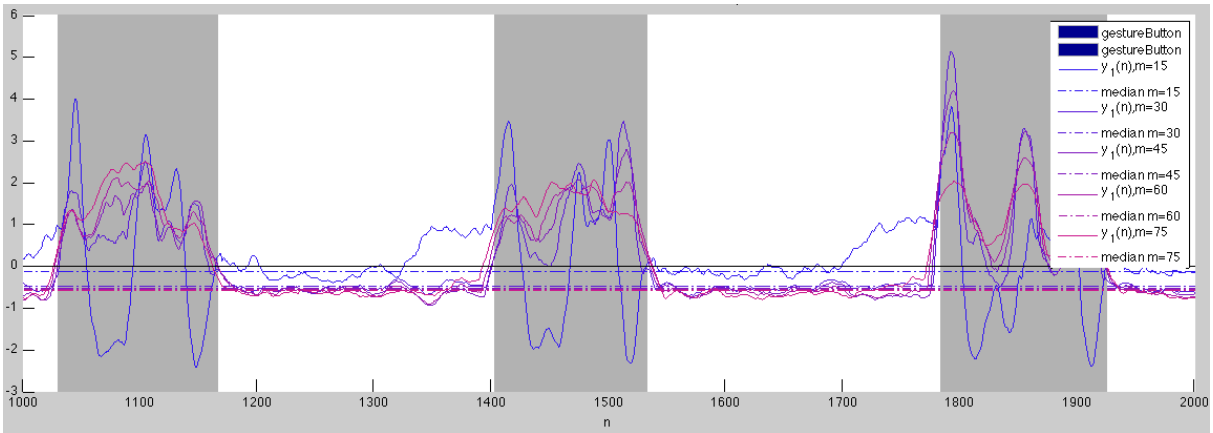

Abb. A.20.: *Segmentierung mit einfacher SFA über Beschleunigungsrohdaten acc_{z_c}: 1.slowest (embedding-dim:15-75) + $Median_c$ (mirrored)*

Abbildungsverzeichnis

2.1. Inter- und Intraclass Varianz für erstes SFA Ausgangssignal 10
2.2. Wii Orientierung 11
2.3. GUI des entwickelten WiiTools zur Gestenaufzeichnung 12
2.4. Beispielausschnitt der aufgezeichnete Gestendaten mit dem WiiTool 13
2.5. Verwendete Gesten: *circle*, *throw*, *frisbee*, *bowling* und z 14
2.6. Längennormierung (Bsp.Kreis z-Beschleunigung) vorher - nachher 17
2.7. Amplitudennormierung (Bsp.Kreis z-Beschleunigung) vorher/nachher 18
2.8. Zeitreihe der Kreis-Geste nach kompletter Vorverarbeitung 18
2.9. Vergleich Bootstrap-Verfahren (personenabhängig) 26
2.10. Vergleich Bootstrap-Verfahren (zwei neue Personen) 27
2.11. Vergleich Bootstrap-Verfahren (neue Personen) über alle Daten 28
2.12. Vergleich der personenabhängigen Klassifizierungen 31
2.13. Vergleich der Klassifizierung von Gesten einer neuen Person 34
2.14. Vergleich Klassifizierung neuer Personen mit SFA mit/ohne PB 37
2.15. Ergebnisse Minimales Trainingsset 38

3.1. Zeitreihen der Beschleunigungsrohdaten (*circle*) 45
3.2. Zeitreihen der Beschleunigungsrohdaten (*throw*) 45
3.3. Zeitreihen der Beschleunigungsrohdaten (*frisbee*) 46
3.4. Zeitreihen der Beschleunigungsrohdaten (*bowling*) 46
3.5. Zeitreihen der Beschleunigungsrohdaten (z) 46
3.6. Auswirkung der Kalibrierung auf r 48
3.7. Regelbasierte Segmentierung 50
3.8. Segmentierung mit SFA über Beschleunigungsbetrag 51
3.9. Segmentierung mit SFA über Beschleunigungsrohdaten (Input/Output) 52
3.10. Segmentierung mit einfacher SFA über Beschleunigungsrohdaten 53
3.11. Segmentierung mit kaskadierter SFA über Beschleunigungsrohdaten 54
3.12. Vergleich Segmentierung (*circle*): SFA über Beschleunigungsbetrag 56
3.13. Vergleich Segmentierung (*circle*): einfache SFA über Beschleunigungsrohdaten . . 56
3.14. Vergleich Segmentierung (*circle*): kaskadierte SFA über Beschleunigungsrohdaten 56
3.15. Vergleich Segmentierung (*throw*): SFA über Beschleunigungsbetrag 57
3.16. Vergleich Segmentierung (*throw*): einfache SFA über Beschleunigungsrohdaten . . 57
3.17. Vergleich Segmentierung (*throw*): kaskadierte SFA über Beschleunigungsrohdaten 57
3.18. Vergleich Segmentierung (*frisbee*): SFA über Beschleunigungsbetrag 58
3.19. Vergleich Segmentierung (*frisbee*): einfache SFA über Beschleunigungsrohdaten . 58
3.20. Vergleich Segmentierung (*frisbee*): kaskadierte SFA über Beschleunigungsrohdaten 58
3.21. Vergleich Segmentierung (*bowling*): SFA über Beschleunigungsbetrag 59
3.22. Vergleich Segmentierung (*bowling*): einfache SFA über Beschleunigungsrohdaten . 59
3.23. Vergleich Segmentierung (*bowling*): kaskadierte SFA über Beschleunigungsrohdaten 59
3.24. Vergleich Segmentierung (z): SFA über Beschleunigungsbetrag 60
3.25. Vergleich Segmentierung (z): einfache SFA über Beschleunigungsrohdaten 60
3.26. Vergleich Segmentierung (z): kaskadierte SFA über Beschleunigungsrohdaten . . . 60
3.27. Segmentierung über durschnittliche Bewegungsänderung (*circle*) 61
3.28. Segmentierung über durschnittliche Bewegungsänderung (*throw*) 62
3.29. Segmentierung über durschnittliche Bewegungsänderung (*frisbee*) 62
3.30. Segmentierung über durschnittliche Bewegungsänderung (*bowling*) 63
3.31. Segmentierung über durschnittliche Bewegungsänderung (z) 63
3.32. Vergleich Segmentierung: Regelbasierte Segmentierung 65

3.33. Vergleich Segmentierung: Segmentierung mit SFA 65
3.34. Vergleich Segmentierung: Segmentierung über Beschleunigungsänderung 65

A.1. Regelbasierte Segmentierung (*circle*) 71
A.2. Regelbasierte Segmentierung (*throw*) 71
A.3. Regelbasierte Segmentierung (*frisbee*) 72
A.4. Regelbasierte Segmentierung (*bowling*) 72
A.5. Regelbasierte Segmentierung (z) 72
A.6. Segmentierung mit SFA über Beschleunigungsbetrag (*circle*) 73
A.7. Segmentierung mit SFA über Beschleunigungsbetrag (*throw*) 73
A.8. Segmentierung mit SFA über Beschleunigungsbetrag (*frisbee*) 74
A.9. Segmentierung mit SFA über Beschleunigungsbetrag (*bowling*) 74
A.10. Segmentierung mit SFA über Beschleunigungsbetrag (z) 74
A.11. Segmentierung mit SFA über Beschleunigungsrohdaten (*circle*) 75
A.12. Segmentierung mit SFA über Beschleunigungsrohdaten (*throw*) 75
A.13. Segmentierung mit SFA über Beschleunigungsrohdaten (*frisbee*) 76
A.14. Segmentierung mit SFA über Beschleunigungsrohdaten (*bowling*) 76
A.15. Segmentierung mit SFA über Beschleunigungsrohdaten (z-Geste) 76
A.16. Segmentierung mit einfacher SFA über Beschleunigungsrohdaten (*circle*) 77
A.17. Segmentierung mit einfacher SFA über Beschleunigungsrohdaten (*throw*) 77
A.18. Segmentierung mit einfacher SFA über Beschleunigungsrohdaten (*frisbee*) 78
A.19. Segmentierung mit einfacher SFA über Beschleunigungsrohdaten (*bowling*) 78
A.20. Segmentierung mit einfacher SFA über Beschleunigungsrohdaten (z) 78

Tabellenverzeichnis

2.1. Überblick über die verwendeten Gestendaten zur Klassifizierung 15
2.2. Überblick Gestendauer pro Geste und pro Person im Durchschnitt (in Sekunden) 16
2.3. SFA CV-Test mit steigender PCA-Reduktionsdimension 23
2.4. Vergleich Bootstrap über 0-600 PB-Pattern (Person 122) 26
2.5. Ergebnisse der personenabhängigen Klassifizierung mit SFA, Gauss, HMM 29
2.6. Konfusionsmatrizen personenabhängige Klassifizierung mit SFA 30
2.7. Konfusionsmatrizen personenabhängige Klassifizierung mit HMM 31
2.8. Ergebnisse personenübergreifenden Klassifizierung 32
2.9. Konfusionsmatrix personenübergreifende Klassifizierung mit SFA 33
2.10. Konfusionsmatrix personenübergreifende Klassifizierung mit HMM 33
2.11. Ergebnisse Klassifizierung einer neuen Person 35
2.12. Konfusionsmatrix Klassifizierung einer neue Person mit SFA 36
2.13. Konfusionsmatrix Klassifizierung einer neue Person mit HMM 36
2.14. Ergebnisse Klassifizierung neuer Personen mit SFA und PB(*noise* 250) 37
2.15. Ergebnisse Minimales Trainingsset 39

3.1. Übersicht verwendete Gestendaten mit Button-Gestenindikator 44
3.2. Gemessene Werte des Aufzeichnungsgerätes 47
3.3. Gemessene Werte eines nicht verwendeten anderen Gerätes 47
3.4. Parameter für regelbasierte Segmentierung (Hysterese) 50
3.5. Segmentierung mit SFA: Verlauf des Medians für unterschiedliche m 53
3.6. Parameter für Segmentierung mit SFA über Beschleunigungsbetrag 55
3.7. Parameter für Segmentierung mit einfacher SFA über Beschleunigungsrohdaten . 55
3.8. Parameter für Segmentierung mit kaskadierter SFA über Beschleunigungsrohdaten 55
3.9. Parameter für Segmentierung über durchschnittliche Bewegungsänderung 61
3.10. Gegenüberstellung der Segmentierungsverfahren 64

A.1. Ergebnisse der einzelnen Testläufe zur personenabhängig Klassifizierung mit HMM 70

Literaturverzeichnis

[Ber03] Pietro Berkes. sfa-tk: Slow Feature Analysis Toolkit for Matlab (v.1.0.1), 2003, Available online at `http://itb.biologie.hu-berlin.de/~berkes/software/sfa-tk/sfa-tk.shtml` visited on 2010-05-02 .

[Ber05] Pietro Berkes. Pattern Recognition with Slow Feature Analysis. *Cognitive Sciences EPrint Archive (CogPrints)*, 4104, 2005, Available online at `http://cogprints.org/4104/`.

[Bre01] Leo Breimann. Random Forests. *Machine Learning*, 2001.

[HS06] Niek Hassink and Maarten Schopman. Gesture recognition in a meeting environment. *Masterthesis, University of Twente, Enschede, Netherlands*, Feb 2006.

[HTF01] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2001.

[Hei09] Kristine Hein. Projektbericht: Lernende Klassifikation beschleunigungsbasierter 3D-Gesten des Wii-Controllers. *University of Applied Sciences Cologne, Gummersbach (cms.f10.fh-koeln.de)*, Jan 2009.

[HHH98] Frank G Hofmann, Peter Heyer, and Günther Hommel. Velocity profile based recognition of dynamic gestures with discrete hidden Markov models. *Gesture and Sign Language in Human-Computer Interaction: International Gesture Workshop, Bielefeld, Germany, September 1997. Proceedings*, Lecture Notes in Computer Science(Volume 1371/1998): pages 81ff, Oct 1998, Available online at `http://www.springerlink.com/content/ql3qwapwyy5u6thl/`.

[Inc06] AiLive Inc. AiLive LiveMove Pro, Oct 2006, Available online at `http://www.ailive.net/liveMovePro.html` visited on 2010-05-05 .

[KKM+06] Juha Kela, Panu Korpipää, Jani Mäntyjärvi, Sanna Kallio, Giuseppe Savino, Luca Jozzo, and Sergio Di Marca. Accelerometer-based gesture control for a design environment. *Personal and Ubiquitous Computing*, 10(5): pages 285–299, Aug 2006.

[Ken72] Adam Kendon. Some Relationships between Body Motion and Speech, in B. Poppe and A. Wolfe, eds., Studies in Dyadic Communication. *Pergamon Press*, pages 177–210, 1972.

[KKH10] Patrick Koch, Wolfgang Konen, and Kristine Hein. Gesture Recognition on Few Training Data using Slow Feature Analysis and Parametric Bootstrap. *2010 International Joint Conference on Neural Networks*, 2010.

[Kon09] Wolfgang Konen. On the numeric stability of the SFA implementation sfa-tk. *Arxiv preprint arXiv:0912.1064*, Dec 2009, Available online at `http://arxiv.org/pdf/0912.1064`.

[LZWV09] Jiayang Liu, Lin Zhong, Jehan Wickramasuriya, and Venu Vasudevan. uWave: Accelerometer-based personalized gesture recognition and its applications. *IEEE Int. Conf. on Pervasive and Mobile Computing*, pages 1–9, 2009.

[MS08] Per Malmestig and Sofie Sundberg. SignWiiver - implementation of sign language technology. *Bachelor Thesis, Universität Göteborg*, Jun 2008, Available online at `http://www.tricomsolutions.com/academic_reports.html`.

[MKKK04] Jani Mäntyjärvi, Juha Kela, Panu Korpipää, and Sanna Kallio. Enabling fast and effortless customisation in accelerometer based gesture interaction. *Proceedings of the 3rd international conference on Mobile and ubiquitous multimedia*, pages 25–31, 2004.

[McN92] David McNeill. Hand and Mind: What Gestures Reveal about Thought. *University of Chicago press*, 1992.

[Neu09] Janosch Gabriel Neuweiler. Modellierung Invarianter Ortserkennung mittels Slow Feature Analysis. *Diplomarbeit Universität Bremen*, Jan 2009, Available online at `http://www.informatik.uni-bremen.de/~josh/Neuweiler2009.pdf`.

[Pop07] Benjamin Poppinga. Beschleunigungsbasierte 3D-Gestenerkennung mit dem Wii-Controller. *Individuelles Projekt, Carl von Ossietzky Universität Oldenburg*, 2007.

[PS07] Benjamin Poppinga and Thomas Schlömer. A Java-based gesture recognition library for the Wii remote. *Carl von Ossietzky Universität Oldenburg*, Aug 2007, Available online at `http://www.wiigee.org`.

[Pre08] Zoltán Prekopcsák. Accelerometer Based Real-Time Gesture Recognition. *POSTER 2008, Prague*, May 2008.

[Pre07] William H. Press. Numerical recipes: the art of scientific computing. *Cambridge University Press*, page 892ff, 2007.

[RBA08] Matthias Rehm, Nikolaus Bee, and Elisabeth André. Wave like an Egyptian: accelerometer based gesture recognition for culture specific interactions. *BCS-HCI '08: Proceedings of the 22nd British HCI Group Annual Conference on HCI 2008: People and Computers XXII: Culture, Creativity, Interaction*, 1, Sep 2008, Available online at `http://portal.acm.org/citation.cfm?id=1531514.1531517`.

[SPHB08] Thomas Schlömer, Benjamin Poppinga, Niels Henze, and Susanne Boll. Gesture recognition with a Wii controller. *TEI '08: Proceedings of the 2nd international conference on Tangible and embedded interaction*, Feb 2008, Available online at `http://portal.acm.org/citation.cfm?id=1347390.1347395`.

[Wis98] Laurenz Wiskott. Learning Invariance Manifolds. *Proc. of the 5th Joint Symposium on Neural Computation, San Diego, CA*, pages 196–203, May 1998, Available online at `http://citeseer.ist.psu.edu/97456`.

[WS02] Laurenz Wiskott and Terrence Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural computation*, 14(4): pages 715–770, Apr 2002.

[Wis03] Laurenz Wiskott. Estimating Driving Forces of Nonstationary Time Series with Slow Feature Analysis. *Arxiv preprint cond-mat/0312317*, Dec 2003, Available online at `http://www.arxiv.org/abs/cond-mat/0312317%20`.

[o.V10] o.V. Wiimote - WiiBrew Calibration EEProm, Apr 2010, Available online at `http://wiibrew.org/wiki/Wiimote` visited on 2010-05-03 .

Erklärung

Ich versichere, die von mir vorgelegte Arbeit selbständig verfasst zu haben. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer entnommen sind, habe ich als entnommen kenntlich gemacht. Sämtliche Quellen und Hilfsmittel, die ich für die Arbeit benutzt habe, sind angegeben. Die Arbeit hat mit gleichem Inhalt bzw. in wesentlichen teilen noch keiner anderen Prüfungsbehörde vorgelegen.

(Ort, Datum) (Unterschrift)